Greening the Switch

Ganesh Ananthanarayanan
Randy H. Katz

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2008-114
http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-114.html

September 10, 2008

Greening the Switch

Ganesh Ananthanarayanan
University of California, Berkeley
ganesha@cs.berkeley.edu

Randy H. Katz
University of California, Berkeley
randy@cs.berkeley.edu

Abstract

Reducing power consumption in the Internet infrastructure is receiving significant attention. We propose three schemes for power reduction in network switches − Time Window Prediction, Power Save Mode and Lightweight Alternative. These schemes are adaptive to changing traffic patterns and automatically tune their parameters to guarantee a bounded and specified increase in latency. We propose a novel architecture for buffering ingress packets using *shadow* ports.

We test our schemes on packet traces obtained from an enterprise network, and evaluate them using realistic power models for the switches. Our simple power reduction schemes produce power savings of 20 to 35% with minimal increase in latency or packet-loss. With appropriate hardware support in the form of Wake-on-Packet, shadow ports and fast transitioning of the ports between its high and low power states, these savings reach 90% of the optimal algorithm's savings.

1 Introduction

The energy efficiency of Internet equipment is important for economic as well as environmental reasons. Networking equipment is experiencing an increase in performance − switches with speeds of 10 Gbps are in the market − that has caused a substantial increase in its power consumption. Studies estimate the USA's network infrastructure uses 24 TWh per year [8, 16], or $24 billion. Environmental concerns have led to standards like EnergyStar and a task force by the IEEE for energy efficient ethernet [4].

Networks are generally provisioned for peak loads due to performance reasons. But network traffic has been observed to have two characteristics - (i) Bursty with long interspersed idle periods [12], and (ii) Diurnal variations in loads, e.g., web servers [21]. This results in heavily under-utilized equipment during non-peak times. Studies have shown that network utilization is under 30% even for backbone networks [5].

The theme of our power reduction schemes is to trade some performance (latency and packet-loss) for reduced power consumption. We propose three schemes − Time Window Prediction, Power Save Mode and Lightweight Alternative. Time Window Prediction (TWP) takes a look at the fine-grained traffic patterns, predicts idle sleep windows and turns the switch ports to low-powered modes during such periods. Power Save Mode (PSM) is similar to the standard in IEEE 802.11 wireless networks [15]. In both the schemes, ports transition into low-powered modes and wake up to process buffered egress packets. Ingress packets are received and processed using *shadow* ports. Both schemes are cognizant of changes in traffic patterns and are constrained by user-specified bounded increase in per-packet latency. Lightweight Alternative addresses over-provisioning of network equipment by providing every high-powered network connection (designed for peak loads) with a corresponding low-powered connection. Only one of the two is appropriately powered-up and used depending on the traffic load.

Our schemes assume some hardware characteristics that enhance power savings. Some of them are already available today and we aim to demonstrate the utility of the rest, and make a strong case for their universal incorporation in future switch designs. Firstly, we assume that switch ports support at least two states - high-powered and low-powered. This is analogous to states in processors that power off sub-components and reduce operational frequency. Wireless network cards also support these concepts. The transition time between states is a crucial factor in performance. Secondly, we require packets to be buffered even when a port is powered down. Modern switches are equipped with buffers in the order of hundreds of kilobytes to a few megabytes [1] to store egress packets. We propose a novel architecture called *shadow* ports for processing ingress packets when a port

is in low-powered mode. Thirdly, switch ports support a Wake-on-Packet (WoP) facility: switch ports that are powered down can be automatically woken up on an incoming packet, with the loss of that packet. While not currently supported, we speculate that the available technology of Wake-on-LAN [7] could be adapted.

We evaluate our power reduction schemes on real-world packet traces collected from an enterprise network. Our schemes produce power savings of upto 20 to 35%. With appropriate hardware support in the form of Wake-on-Packet and fast transitioning of the ports between its high and low power states, these power savings are 90% of the savings achieved by the optimal algorithm. Overall, we make the following contributions: Firstly, we present two simple power reduction schemes − Time Window Prediction and Power Save Mode − that we believe are easy to implement on switches. The schemes operate stand-alone on a switch and hence can be incrementally deployed. Secondly, we analyse the trade-off between performance and power consumption. In doing so, we introduce and demonstrate the value of pre-specified and bounded performance degradation. Finally, we make a set of recommendations for switch designs viz., lightweight alternative, shadow ports, Wake-on-Packet and low-powered modes in switch ports with fast transitioning.

The rest of the paper is organized as follows. Section 2 presents an overview of the power reduction schemes, a detailed description of the switch's power model and evaluation methodology. Sections 3 and 4 describe the Time Window Prediction and Power Save Mode schemes respectively. We describe the Lightweight Alternative in Section 5. Section 6 discusses additional issues. Related work is presented and contrasted with our work in Section 7. We conclude in Section 8.

2 Models and Overview

In this section, we present the architecture of the switch including the power model and the buffering capabilities, overview of the power reduction schemes and evaluation methodology.

2.1 Switch Model

A typical modular switch's power consumption is divided among four main components − chassis, switching fabric, line-cards and ports. The chassis includes the cooling equipment, e.g., fan, among other things and its power consumption is denoted as $Power_{fixed}$. The switching fabric is responsible for learning and maintaining the switching tables and its power consumption is denoted as $Power_{fabric}$. The line-card maintains buffers for storing packets. Ports contain the networking circuitry. The line-card acts as a backplane for multiple ports and forwards packets between the switching fabric and ports. Modern line-cards can support 24, 48 or 96 ports. Note that a switch can contain multiple line-cards. We denote the line-card's power consumption to be $Power_{line-card}$ and every port's power consumption to be $Power_{port}$. Hence the total power consumption of the switch can be viewed as, $Power_{switch} = Power_{fixed} + Power_{fabric} + numLine * Power_{line-card} + numPort * Power_{port}$.

Parameter	Value
$Power_{fixed}$	60W
$Power_{fabric}$	315W
$Power_{line-card}$ (first card)	315W
$Power_{line-card}$ (subsequent cards)	49W
$Power_{port}$	3W
$Power_{port}$ (idle)	0.1W
Port-transition − Power	2W
Port-transition − Time (δ)	1ms to 10ms

Table 1: **Parameters used in the Power Model and their values**

We use values for $Power_{fixed}$, $Power_{fabric}$ and $Power_{line-card}$ from the Cisco Power Calculator [2] corresponding to the Catalyst 6500 switch (we discuss $Power_{port}$ shortly). We assume that the power consumed by this switch is indicative and typical of similar products. Table 1 lists the parameters in our power model and their values. TWP and PSM schemes concentrate on intelligently putting ports to sleep during idle periods; other components of the switch are assumed to be powered on always. Hence, for a switch with say four line-cards and 192 ports, the maximum power saving works out to 39.4%. If all the ports are in low-powered state for 100% of the time, the power savings will be 39.4%. This is the upper-bound for power savings that can be achieved by any scheme that focuses on powering down ports.

2.1.1 Port Design

We define an intuitive two-state Markov model for a port's power states. Ports transition between the high-power and low-power states (see Figure 1). Our power reduction schemes use different criteria for intelligently transitioning the ports between the two states. The transition is assumed to take a finite time δ and is timer-driven − transitions occur after pre-fixed times.

Every port consumes 3W in its high-power state [3]. Consistent with prior work [20], for values of transition time δ, and power-consumption in low-power state, we use values from the wireless domain. We assume the

power consumed in waking up a port to be 2W with a transition time δ of 10 ms. The port consumes 0.1W in low-power state. These values are listed in Table 1.

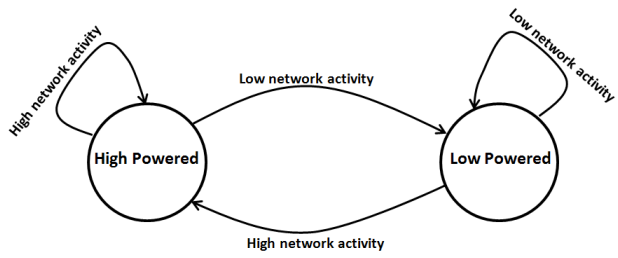

Figure 1: **Markov model for a port's states**

Buffering: Each port is bi-directional and has packets flowing in the ingress – into the switch – and egress – away from the switch – directions. Egress packets are buffered automatically when the port is in low-power state. When a port transitions back to high-powered state, it processes the buffered packets. Ingress packets, on the other hand, have to be received by the port and forwarded to the buffers for further processing. With current switches, ingress packets that arrive when a port is not in high-power state are lost. To address this issue, we propose the idea of a *shadow* port. A shadow port receives ingress packets if any of the normal ports are in low-power state. A shadow port's hardware is expected to be similar to normal ports.

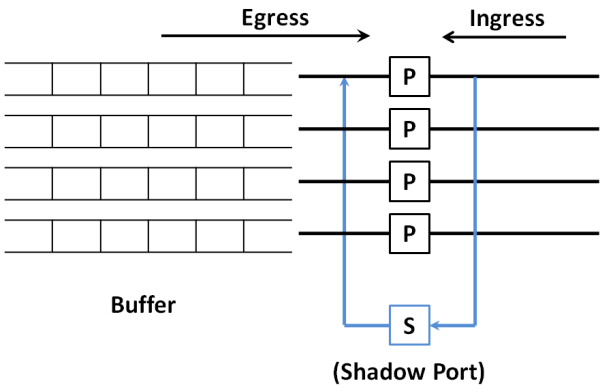

Figure 2: **Architecture of Shadow Port for a cluster of size 4**

Each shadow port is mapped to a cluster of normal ports. Since a shadow port's power consumption is the same as a normal port, savings can be achieved only if atleast two of the normal ports in a cluster are in low-power state in the same time. Figure 2 shows a conceptual diagram of the shadow port's architecture for a cluster of size 4. We assume the presence of some mechanism to distinguish between egress and ingress packets in the buffer. Shadow ports would be able to receive only one incoming packet at a time. If packets arrive simultaneously for different normal ports that are in low-powered state, all but one of them are lost. While ingress packets are lost due to simultaneous arrivals on the shadow port, egress packets are lost due to buffer overflow. We analyze packet-loss and its dependency on the cluster and buffer sizes in our evaluations of the power reduction schemes.

Wake-on-Packet: Note that a port's transition between its power states in Figure 1 incurs an overhead of power and time. During sustained idle periods, to avoid the overhead of timer-driven transitions to high-powered state, we propose a Wake-on-Packet (WoP) facility. Using this facility, a port automatically transitions from the low-powered state to the high-powered state on arrival of a packet. This packet is lost if it is an ingress packet; egress packets are all buffered. Note that shadow ports do not receive ingress packets for a port that has put itself to indefinite sleep relying on WoP.

While timer-driven transitions are still valuable as they enforce a minimum sleeping duration even in the presence of traffic flow, WoP helps the ports take better advantage of sustained idle periods. The hardware support for this capability would be similar to Wake-on-LAN [7].

2.2 Power Reduction Schemes – Overview

We present three power reduction schemes based on a combination of predicting low-activity periods in traffic flow and trading a bounded increase in latency for power reduction. We consider sleep opportunities at the level of individual ports.

1. *Time Window Prediction* (TWP) observes the number of packets crossing a port in a time window and assumes that to be an indicator of the network activity in the next window. If the number of packets in that window is less than a threshold (or zero), the port is powered down for the next time window.

2. *Power Save Mode* (PSM) is similar to IEEE 802.11 networks wherein the Access Point buffers the packets while the wireless network card of the client is powered down [15]. In our Power Save Mode, the switch buffers the packets (refer Section 2.1.1) while ports remain powered down.

   Both the Time Window Prediction scheme and Power Save Mode are adaptive and automatically configure their operational parameters to guarantee a pre-specified and limited increase in per-packet latency. We assume this bound can be specified at the level of individual ports using SNMP MIBs. Specifying it at the port-level is useful when servers with

varied application characteristics are connected to the same switch, e.g., the tolerable increase in latency is likely to be different for web servers and streaming servers.

3. *Lightweight Alternative* strategy is an attempt to balance the over-provisioning in networks to handle peak loads. While these high-powered modular switches are useful during peak loads, they remain under-utilized otherwise. Activity in servers has been observed to have a diurnal variation and our suggestion is to have low-power alternatives to serve low-activity periods. Every high-powered switch has a low-powered alternative – a low-powered integrated switch or wireless access point substituting the high-powered modular switch. All machines have connectivity through the high-powered switch as well as its low-powered alternative. Using an appropriate learning algorithm the system automatically identifies low-activity "slots". During low-activity slots, the high-powered switches are powered down and communication happens through the low-powered alternatives. We show that the economic overhead of adding new hardware is compensated by the savings in power.

The time periods in the Time Window Prediction scheme and Power Save Mode are of the order of milliseconds to seconds while the slots in Lightweight Alternative scheme are expected to be in minutes.

2.3 Evaluation Methodology

Now we present our evaluation procedure. Our figures of merit are the percentage of power reduced as well as the increase in latency and packet loss. Our baseline power consumption assumes all the switch's components to be powered up throughout. We calculate the reduction in $Power_{switch}$ because of our schemes. Prior work [22, 19] has used the percentage of times when the port is in a low-power state as a metric for evaluation. While such a metric is useful to analyze the efficacy of the scheme, evaluating the overall power reduction is much more indicative because powering down a fraction of ports does not result in the power consumption of other components like the switching fabric, line-cards and chassis being zero. Overall power savings is our primary metric for evaluation. To summarise, we perform the following evaluations on TWP and PSM:

1. Power Savings
2. Latency and Packet Loss for varying buffer sizes
3. Effect of Wake-on-Packet and transition time δ on the power savings
4. Effect of cluster size on Power Savings and Packet Loss

2.3.1 Traces

We evaluate traces from a Fortune 500 company's enterprise network of PC clients and file and other servers[1]. We evaluate our schemes for varying and low latency bounds and a more comprehensive evaluation with datacenter traffic patterns is part of future work.

Our enterprise traces were collected in the Fortune 500 company's LAN in March 2008 for a period of 7 days. We collected SNMP MIB counter data of the number of packets across every port (ingress and egress measured separately) on a switch with four line cards (192 ports). This counter data was collected once in every 20 seconds. Consistent with previous studies [23], we assume a Pareto distribution of packets within the 20 second interval. The Pareto distribution captures the bursty nature of traffic.

2.3.2 Optimal Power Reduction

We put our results in context by comparing them with the optimal power reduction scheme. The optimal scheme assumes an *oracle* that exactly knows the inter-arrival times of the packets and hence can put the switch in a low-power mode during each of its idle periods. The optimal scheme assumes an instantaneous transitioning between the power states of the switch in Figure 1. With our traces and power model, the optimal power saving is 33.9% implying an utilization of 16.8%.

3 Time Window Prediction

In this section, we describe the Time Window Prediction scheme and evaluate its performance.

3.1 Scheme Definition

This scheme observes the number of packets crossing a port in a sliding time-window of t_o units and assumes that to be an indication of traffic in the next window. If the number of packets in this time-window is below a threshold τ_p, the switch powers down the port for t_s units (sleep time window). In other words, it makes a transition between the states in Figure 1 depending on the observation in this time window. Packets that arrive at the port when it is powered down are buffered (refer Section 2.1.1).

[1]Due to the non-disclosure clause we cannot disclose the name of the company

Table 2: **Time Window Prediction**

Inputs:
- Packet Threshold for sleeping: τ_p
- Size of the observation time window: t_o
- Size of the sleep time window: t_s
- Lower bound for sleeping: τ_s
- Bound for increase in latency: L

Variables:
- $noSleep$ = false
- Average long-term per-packet increase in latency: *avg-lat* = 0

Step 1: Count number of packets, *num-packet*, in t_o window.
Step 2: If *num-packet* $< \tau_p$
- If *noSleep* is false
  - Sleep for t_s window
  - Process buffered packets
  - Calculate the average increase in per-packet latency for the packets that arrived in the $(t_o + t_s)$ window, $delay_{recent}$
- Else
  - Calculate the average increase in per-packet latency for the packets that arrived in the (t_o) window, $delay_{recent}$
- *weighted-latency* = $w_1 *$ *avg-lat* + $w_2 * delay_{recent}$
- *adapt-ratio* = *weighted-latency* / L
- If (t_s / *adapt-ratio* $> \tau_s$)
  - $t_s \leftarrow t_s$ / *adapt-ratio*
- Else
  - $noSleep$ = true

Step 3: Update *avg-lat*
Step 4: Go to Step 1

Adaptive Sleep Window: The basic Time Window Prediction scheme is not cognizant of performance degradation. While a good prediction function would reduce erroneous sleeps and the consequent increase in latencies, we believe that the performance of the scheme should not entirely rely on the accuracy of the prediction function. Egress packets that arrive at a port when it is asleep are buffered and sent after the port wakes up. This causes an increase in latency. Note that this in addition to the latency incurred due to various factors along its path. For the sake of brevity, we interchangeably refer to this increase in latency as simply latency. Ingress packets are handled by the shadow port and incur no extra latency.

We augment the basic TWP scheme by making it sensitive to packet latencies and offer latency guarantees. TWP is supplied with a per-port bound on the tolerable increase in per packet latency, and it dynamically adapts its sleep-window t_s to meet the latency bounds. Note that t_s is also automatically increased in times of low network activity and this increases the power savings. Table 2 describes the adaptive Time Window Prediction scheme.

Latency of a packet can be calculated using its time of arrival and time of processing. *avg-lat* is the running average for the per-packet increase in latency over a long term. The weights w_1 and w_2 were each set to 0.5 for the evaluations. The motivation is to ensure that the scheme is adequately sensitive to changes in traffic patterns. The lower-bound for sleeping τ_s is set to twice the transition time δ to ensure that the overhead of powerind down and waking up a port is not higher than the power saving because of the sleep.

TWP does not look across multiple observation time windows. The size of the sleep window, and hence ratio of the observation time window to the sleep window is adaptively adjusted. This is equivalent to observing across multiple time windows.

Wake-on-Packet: Ports periodically wake up at the end of the sleep-window in anticipation of sending buffered egress packets. During sustained idle periods, the energy expended due to periodically waking up and

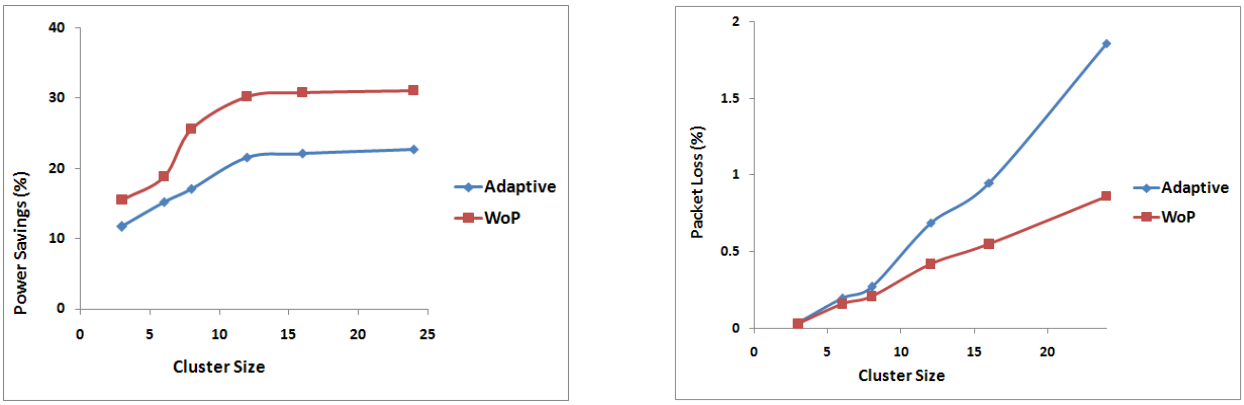

Figure 3: **Variation of power savings (left) and packet loss (right) with cluster size, in Time Window Prediction**

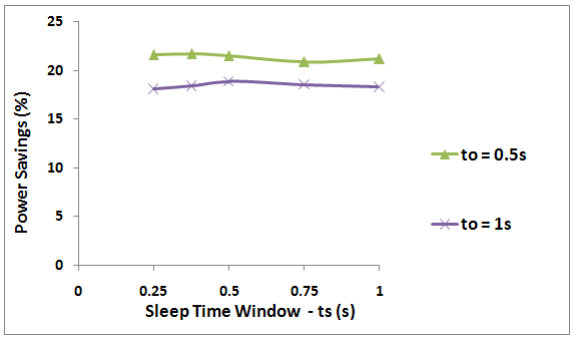

Figure 4: **Power Savings with Time Window Prediction is independent of the sleep time window (t_s). This is because the sleep time is automatically adapted to meet the latency bounds. The independence holds with the Wake-on-Packet capability**

staying awake for t_o units before powering down, is significant and wasted. Even though our scheme is adaptive, it takes a significant amount of time to enlarge the sleep window depending on the value of the weights w_1 and w_2.

We measured the results of our algorithm if the ports supported a Wake-on-Packet (WoP) facility. If there are no packets in multiple consecutive t_o windows, the port can put itself into indefinite sleep and wake up on an incoming packet. Our results in the next section present a strong case to incorporate the Wake-on-Packet facility in future switch designs.

3.2 Evaluation

We evaluate the Time Window Prediction scheme for power savings as well as latency and packet loss.

Cluster Size: As mentioned in Section 2, a cluster of ports are mapped to a shadow port for receiving ingress packets when they are in low-powered state.

t_o	Adaptive	WoP	Increase
0.5s	21.6%	27.3%	27%
1s	18.1%	24.4%	34%

Table 3: **Wake-on-Packet produces a significant increase in power savings in the Time Window Prediction scheme**

Higher number of ports in a cluster will result in a higher probability of multiple ports in the cluster being in low-powered state at the same time, and hence savings in power. Figure 3 plots how the power savings vary with increasing cluster size. But as the figure on the right in Figure 3 shows, higher cluster sizes also result in packet losses. We plot the packet loss only for ingress packets as loss of egress packets is not affected by the cluster size. Based on the graph, we pick a cluster size of 12 for our experiments.

Power Savings: TWP automatically adapts its sleep window, t_s, to meet the latency bounds. As shown in Figure 4, the power savings is a function of only t_o. For a fixed value of t_o, we experimented with varying initial values of t_s – 0.25s, 0.375s, 0.5s, 0.75s and 1s. The results illustrate the adaptive nature of the algorithm whereby the initial value of t_s is automatically and continuously modified to meet the latency bounds.

Table 3 illustrates the benefits of the Wake-on-Packet capability. Note that the power savings achieved with the Wake-on-Packet facility is 80% of the optimal power savings. While we have evaluated our schemes with a transition time 10 ms, we also perform a sensitivity analysis to see the effect of varying δ on the power savings. As shown in Figure 5 the power savings of TWP tend closer to the optimal value for low values of δ. If improvements in hardware facilitate a 1ms transition time, our savings are 90% of the optimal value.

Latency: We evaluate the power savings achieved for

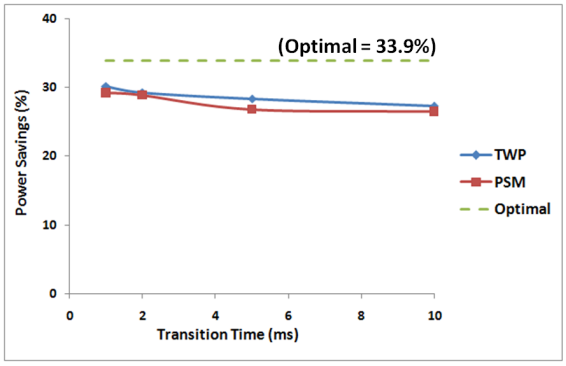

Figure 5: **Effect of the transition time δ on the power savings**

varying latency bounds, L. As shown in Figure 6, the power savings linearly increase with L. As shown in the algorithm in Table 2, an increase in the value of L causes an increase in the *adapt-ratio* which in turn increases the value of the sleep-window t_s. Tolerating higher latencies results in increased sleep durations and hence more power savings.

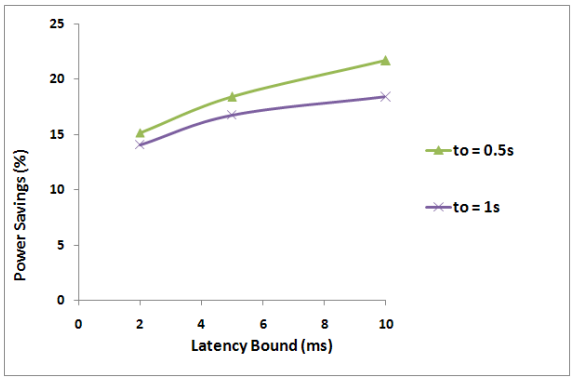

Figure 6: **Power Savings vs. Latency Bound, L, for Time Window Prediction scheme. The trend is is similar in the presence of the Wake on Packet capability**

Packet Loss: Figures 7 plots the packet loss for varying buffer sizes, for the adaptive TWP scheme and how it reduces in the presence of Wake-on-Packet. We note an exponential decay in the packet-losses as the buffer size increases. Packet losses reduce in the presence of WoP. During prolonged idle periods, the adaptive scheme automatically increases its sleep window and this is likely to cause packet losses when packet flow resumes at higher rates because it would take time to shrink the time window. But with WoP, the sleep window remains constant during the indefinite sleep. Both the curves show a packet-loss of under 1% for buffer sizes greater than 500 KB. Most modern switches support buffers in excess of this [1] and hence we believe that the Time Window Prediction scheme produces acceptable packet-losses.

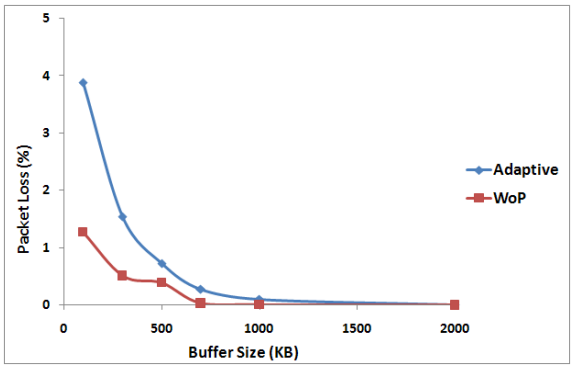

Figure 7: **Packet Loss for varying buffer sizes for Time Window Prediction scheme**

4 Power Save Mode

Now we present the Power Save Mode.

4.1 Scheme Definition

The Power Save Mode (PSM) is primarily based on the switch's capability to buffer packets while the port is powered down. In Power Save Mode, a port periodically goes into low-power mode while the packets are buffered (refer Section 2.1.1). This mode is similar to IEEE 802.11 networks where the client's wireless card powers itself down and the Access Point buffers the packets [15]. The motivation for a port to go to PSM is to reduce power consumption while incurring tolerable performance losses.

Every port cycles between awake (t_a units) and asleep (t_s units). The transitions in Figure 1 are not based on the traffic patterns but rather the size of the awake and asleep time windows. This is where PSM is different from the Time Window Prediction scheme: the sleep in PSM happens with regularity and is not dependent on the traffic flow. While the Time Window Prediction scheme, in the presence of an accurate prediction function, does not necessarily increase latencies, PSM is more aggressive and is naturally expected to cause an increase in latency. PSM is more of a policy decision about powering down ports as opposed to the Time Window Prediction scheme that takes a fine-grained look at the packet arrivals.

Adaptive Sleep Window: Because of its aggressive nature and independence from the traffic patterns, it is all the more important that the sleep window, t_s is automatically adapted in PSM. The adaptation of the sleep window is similar to the adaptive mode in TWP. The sleep

Table 4: **Power Save Mode**

Inputs:
Time awake: t_a
Sleep time window: t_s
Lower bound for sleeping: τ_s
Bound for increase in latency: L

Variables:
$noSleep$ = false
Average long-term per-packet increase in latency: *avg-lat* = 0

Step 1: Stay awake for t_a window
Step 2: If *noSleep* is false
Sleep for t_s window
Process buffered packets
Calculate the average increase in per-packet latency for the packets that arrived in the $(t_a + t_s)$ window, $delay_{recent}$
Else
Calculate the average increase in per-packet latency for the packets that arrived in the (t_a) window, $delay_{recent}$
weighted-latency = w_1* avg-lat* + w_2* $delay_{recent}$
adapt-ratio = *weighted-latency* / L
If (t_s / *adapt-ratio* $> \tau_s$)
$t_s \leftarrow t_s$ / *adapt-ratio*
Else
$noSleep$ = true
Step 3: Update *avg-lat*
Step 4: Go to Step 1

window, t_s is appropriately scaled to meet the latency bound, L. Table 4 describes the Adaptive Power Save Mode. The value of τ_s is set to twice δ, and w_1 and w_2 are set to 0.5 each.

Wake-on-Packet: Similar to the Time Window Prediction scheme, during sustained idle periods, the PSM mode puts the port into an indefinite sleep, only to be woken up by a packet. If there are no packets during multiple t_a windows, the port is indefinitely put to sleep.

4.2 Evaluation

This section presents the evaluation of the Power Save Mode.

Cluster Size: Picking the right cluster size is crucial in producing limited loss of ingress packets. As shown in Figure 8, the trend of increasing power savings and packet losses is the similar to Time Window Prediction scheme. We pick a cluster size of 12.

Power Savings: The adaptive nature of the algorithm is illustrated by the independence of the power savings to the initial value of the the sleep window, t_s. t_s is appropriately modified to meet the latency bounds. We set the initial value of t_s to be 0.25s, 0.375s, 0.5s, 0.75s and 1s. Figure 9 shows the power savings achieved by the adaptive PSM.

t_a	Adaptive	WoP	Increase
0.5s	19.8%	26.5%	33%
1s	12.2%	19.3%	58%

Table 5: **Wake-on-Packet produces a significant increase in power savings in the Power Save Mode**

Table 5 shows the increase in power savings because of the Wake on Packet capability. These achieve 78% of the optimal power savings. As shown in Figure 5 this value increases to 86% for low values of δ. The significant improvement in power savings because of reduced transition times and the Wake on Packet facility is a strong case for incorporation of these features in future switch designs (see Section 6.2).

Latency: Varying latency bounds result in different power savings. The general observation is that true to in-

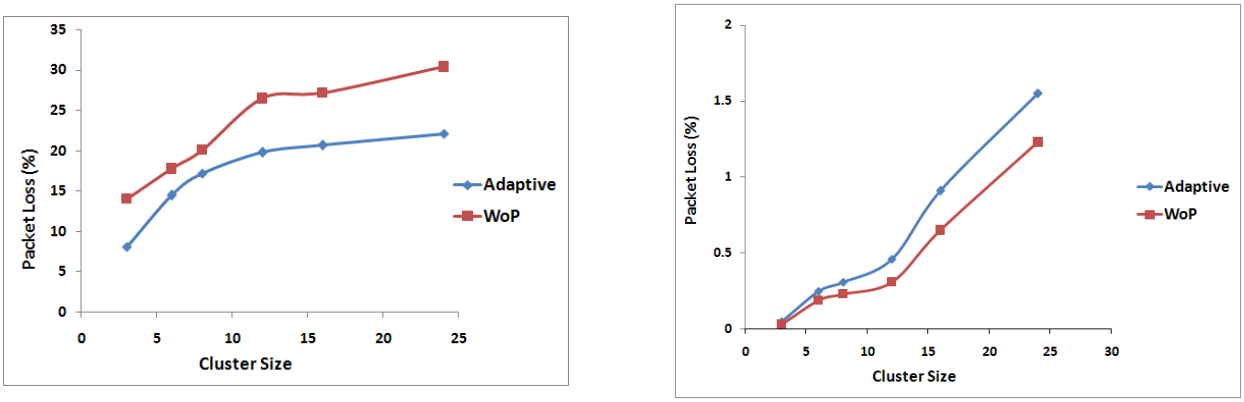

Figure 8: **Variation of power savings (left) and packet loss (right) with cluster size, in Power Save Mode**

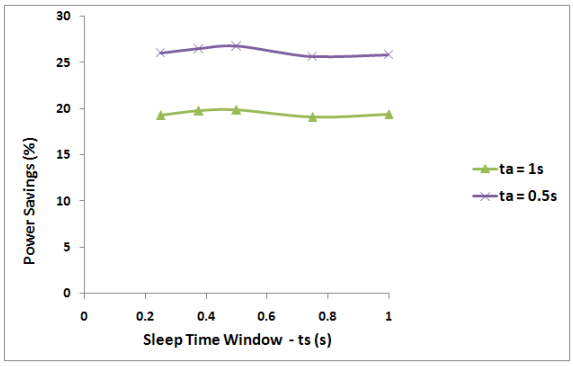

Figure 9: **Power Savings with Power Save Mode is independent of the sleep time window (t_s). The independence holds in the presence of the Wake-on-Packet capability**

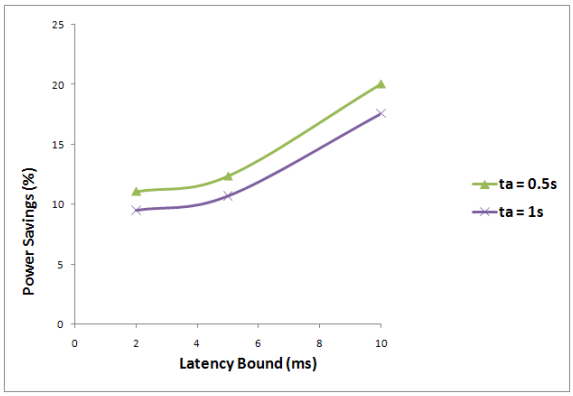

Figure 10: **Power Savings vs. Latency Bound, L, for Power Save Mode. The trend is is similar in the presence of the Wake on Packet capability**

tuition, higher tolerance in latency result in larger power savings. Figure 10 plots the relation. The explanation for the increase in power savings for higher values of L is similar to the Time Window Prediction scheme.

Packet Loss: As in the Time Window Prediction scheme, the packet losses are reduced in the presence of the Wake on Packet capability. Figure 11 plots the results.

4.3 Latency Bound

The value of the latency bound, L is an important input for the PSM and TWP. As shown in the results, the amount of power savings is directly proportional to L (see Figures 6 and 10). The latency bound L is likely to depend on the application type. For example, streaming applications, web browsing and search queries, and file transfers have different latency requirements and the value of L would vary accordingly. In out experiments we tested for L values of 2ms, 5ms and 10ms. While we believe these to be reasonably diverse and constrained values for L, we think appropriately identifying its value based on the application protocol and the packets flowing through a switch is the right approach. Such identification is part of future work.

5 Lightweight Alternative

This scheme attempts to address the problem of over-provisioning in network designs. In contrast to the earlier two schemes, this looks at the macroscopic view of the traffic on the switch. Also, the Time Window Prediction and Power Save Mode algorithms affect only the power consumed by the ports and hence this puts an upper-bound on the amount of savings possible. Since the line-card can be powered down only in the window when all its ports are powered down and switching fabric can be powered down only when all the line-cards are powered down, this can result in the line-cards and switching fabric often being powered on and under-utilized.

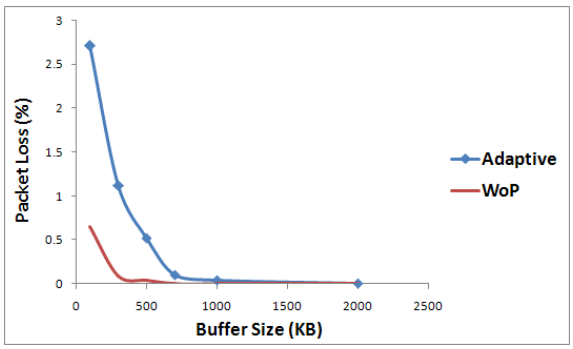

Figure 11: **Packet Loss for varying buffer sizes for the Power Save Mode**

As observed in prior work [21, 10], traffic patterns have a clear diurnal variation. The traffic resembles a sine curve that peaks in the day and experiences a lull in the nights. For instance, enterprise networks can expect to have far fewer users at 3AM compared to 3PM. Similarly, most traffic routed to datacenters in the USA is likely to be from users in North America and it can be reasonably expected that there are fewer users in the night. But most networks are designed for the peak-load and result in idling of resources otherwise.

Our proposal is to deploy low-power or *lightweight* alternatives for every high-powered switch. The high-powered switches are typically modular switches that have multiple line cards and a supervisor engine. These switches support very high packet processing speeds (in the order of Mega packets per second) and have multiple line cards with each of the cards connecting up to 96 machines through its ports. We interchangeably refer to high-powered switches as modular switches. The lightweight alternatives could be low-powered integrated switches or Access Points substituting for the high-powered switch. The lightweight alternatives are expected to have lower packet processing speed and line speeds. We explore these design options in Section 5.2. All machines have connectivity through the high-powered switch as well as the lightweight alternative. From the traffic patterns, the system automatically identifies "slots" of low activity and ensure that only one of the two connections is appropriately powered-up and used depending on the traffic load.

5.1 Low Activity Slots

The system uses the simple K-Means clustering algorithm to identify slots of low activity. The number of packets across days is split into equal-sized slots and the number of packets per slot is logged for t training days. This data is classified using K-Means clustering, with K = 2, to produce two clusters one each for high and low activities. If the difference between the averages of the two clusters is above a threshold, then the slots of low activity can be served by the lightweight alternative. The check for the difference of the averages being higher than a threshold is done to guard against the case where the modular switch is heavily loaded uniformly. The threshold is set depending on the packet processing speed and network bandwidth of the lightweight alternative to ensure minimal performance degradation.

The clustered data is processed to find the count of the total number of days when a particular slot is in the low-activity cluster. If the fraction of days in which a particular slot is in the low activity hour is higher than a confidence level C, then that slot is marked as a low-activity slot. All low-activity slots are served using the lightweight alternative.

The size of the slot is dynamically decided and its lower-bound is set to a minimum value (fixed as 15 minutes in our experiments). We split the training logs of t days into two equal parts - $training_{learn}$ and $training_{test}$. For varying slot sizes, the $training_{learn}$ data is clustered to get low-activity slots and the accuracy of the prediction is tested using the $training_{test}$. We define the prediction to be erroneous if a slot is classified as low-activity but its number of requests is closer to the centroid of the high-activity slot. The slot size with the highest accuracy is picked.

The clustering algorithm can be run either on a static or dynamic training set. In the static model, the training is done once to identify the low-activity slots whereas the dynamic model has a sliding training window using data from the last set of t days for identifying the low-activity slots.

5.2 Design of Lightweight Alternatives

In this section, we explore potential design options for the lightweight alternative. Conceptually, we propose having a lightweight counterpart for every line card in the high-powered switch. The design options discuss the trade-off between deployability and performance with the lightweight alternative, and are by no means exhaustive.

(i) Lightweight Switch: This proposes a re-design of the network topology wherein each line card can be substituted by a separate lightweight switch. The lightweight switches could be integrated or stackable switches (colloquially referred to as pizza box switches). These switches support lower speeds (10/100 Mbps) and have reduced packet processing speeds. Each machine connected to the high-powered switch is also connected to the lightweight switch with only one of the two being active at any given point in time.

The routing tables and other configuration informa-

tion for the lightweight switch can be transferred from the main switch using standard protocols like GARP VLAN Registration protocol (GVRP). GVRP is a protocol for switches to transfer configuration information among each other.

(ii) Wireless: The lightweight alternative connectivity can be provided through a wireless Access Point. This design has the advantage of not having to re-wire the network and can work with standard Access Points. The downside to this scheme is the inherent unreliability and limited bandwidth of wireless links.

Costs: The Lightweight Alternative scheme proposes adding extra hardware in the network and the economic overhead in procuring them is not negligible. But as we show in Section 6.1, a 30% power reduction translates to an economic saving of $37,133 in one year and even a 20% saving leads to a cost reduction of $24,755 in an year. The economic benefits obtained by the power savings are clearly higher than the price of the lightweight alternatives and hence the schemes ensure that cost of the extra hardware is compensated.

5.3 Evaluation

The lightweight alternative scheme relies on clustering to predict low-activity periods and errors in this prediction has repercussions on the performance of the network. We evaluate how the confidence of the classification C affects the prediction error as well as the power savings for both the fixed and dynamic training models.

Prediction Error: The clustering algorithm generates two clusters from the training data, one each for the high and low activity slots. Each cluster has its respective centroid from the clustering algorithm. For evaluation, we define the prediction to be erroneous if a slot is classified as low-activity but its number of requests is closer to the centroid of the high-activity slot. Note that this evaluation is different from the procedure used to identify the right slot size. Figure 12 plots the prediction error for varying confidence of the classification. Dynamic training window increases the prediction accuracy.

Power Savings: The confidence in the classification algorithm affects the power savings. Higher confidence values imply a highly conservative scheme with low prediction errors but low power savings. We illustrate this in Figure 13 (we use the power values corresponding to the Cisco 3500 series [2] for the lightweight alternative). We also see that the dynamic training model is able to better adapt to changing traffic patterns and produces more accurate classifications and higher power savings.

The modular switches have multiple line cards and a significant overhead power cost. Fixing multiple line cards helps amortize this overhead and this is reflected in the lower power savings. As shown in Figure 14, the scope for power savings reduces with increase in the number of line cards.

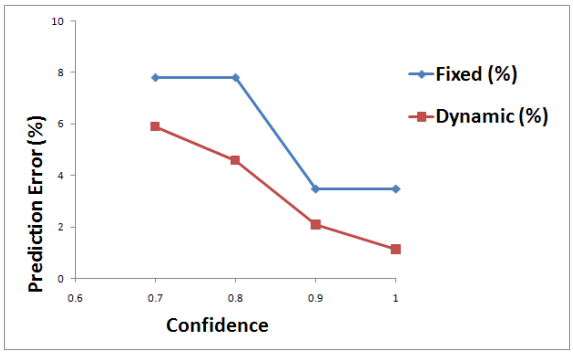

Figure 12: **Prediction Error reduces as the Confidence of classification increases**

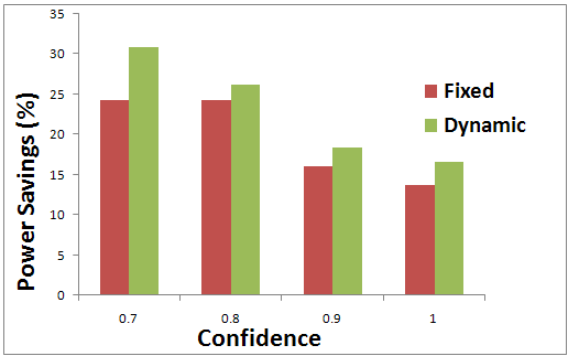

Figure 13: **Power Savings as a function of the Confidence of classification**

Combining TWP and PSM: We also measured the power savings when the Lightweight Alternative works in tandem with the Time Window Prediction or Power Save Mode. One of the high-powered switch or the lightweight alternative is powered up depending on the prediction for the slot, and each of the switches in turn employ either the Time Window Prediction or Power Save Mode. We assume WoP, port-transition time of 10ms and a latency bound of 10ms for TWP and PSM, and a confidence of 0.8 for the lightweight alternative. We see that the power savings obtained using the Lightweight Alternative together with the Time Window Prediction and Power Save Mode are 36% and 34% respectively – higher than the individual power savings of these schemes. Table 6 lists the best-case power savings obtained by TWP and PSM individually, and their comparison to the power savings when combined with Lightweight Alternative.

The power savings do not exactly add up because the individual power savings produced by the TWP and PSM are lower than the values presented earlier. The heavyweight switch handles high-activity slots and the

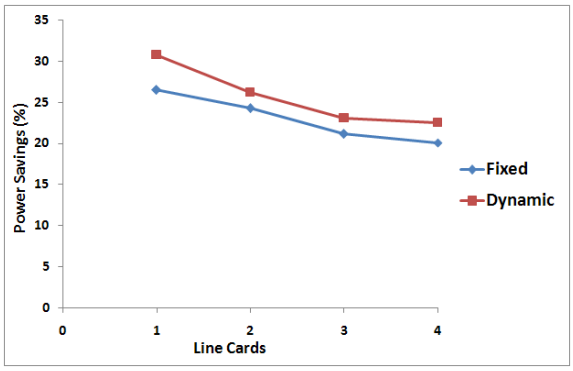

Figure 14: **Power Savings with Lightweight Alternative scheme reduces with increasing number of line cards on the modular switch**

	Individual Savings	Combination
TWP	27.3%	36%
PSM	26.5%	34%

Table 6: **Power Savings when Lightweight Alternative is deployed in tandem with TWP and PSM**

lightweight switch has lower packet processing speed that curtails its opportunity to sleep.

6 Discussion

6.1 Quantifying the Power Savings

In this section we quantify the power savings in equivalent dollars to highlight the value of our power reduction. An average of 30% power savings corresponds to 305.2 kWh over a 30-day period. With an electricity price of 10 cents/kWh, we see that the power savings lead to a reduction of $3,052 in costs for a single switch over the 30-day period. The savings when calculated for an year is $37,133 for a single switch. The overall savings for an organization would depend on the exact nature of the workloads and the switches that are deployed.

6.2 Design Recommendations for Switches

As seen in the evaluation of the Time Window Period and Power Save Mode schemes, there is promise for significant power improvement if the switching hardware were to be cooperative. With the increasing interest among manufacturers of networking equipment to design power efficient switches, our results underline the value of the Wake-on-Packet facility, shadow ports and low transition times between the power states.

Shadow ports enable packets to be received and processed even when a port is in low-powered state. Such a facility effectively ensures that packets in both directions can be buffered even when a port is in low-powered state and will be crucial towards the performance of power reduction schemes.

The Wake-on-Packet facility enables a port to wake up on an incoming packet without losing that packet. This is analogous to the Wake-on-LAN [7] and Wake-on-Wireless [14] ideas that have been prototyped as well as commercially available. Our results strongly advocate the case for such capabilities in switches.

The transition time between the power states in a switch is crucial to most schemes that aim to power down the port during idle periods. Since most schemes tend to be conservative to ensure minimal increase in latency and packet-loss, low transition times would help the schemes be confident about the responsiveness of the switch and thereby present more opportunities for sleeping.

6.3 Traces – Need for benchmarks

Collection of useful, datacenter level traces remain a challenge. We certainly are not the first research project to face this issue and like others, we strongly believe that it is important to develop a set of benchmark traces for evaluating power-reduction schemes. These benchmarks should be developed for the different typical environments – web servers, enterprise networks and datacenters. Such a set is common in the other areas of computer science like computer architecture, e.g., SPEC suite [6]. We believe that such benchmarks would help compare the results of different proposed power-save algorithms. We could not compare our results quantitatively with prior work since we use different traces and these could not be exchanged because of non-disclosure agreements.

7 Related Work

Power conservation in computer networks has been addressed in multiple contexts: power provisioning and load balancing in data centers [10, 25], network switches [17, 20, 13, 22], wireless networks [14, 15, 12, 11] and mobile phones [9, 24].

Wireless networks traditionally focus on power conservation. Wake-on-wireless [14] and Cell2Notify [11] propose on-demand powering up of the wireless interface using intelligent signaling. This ensures that the interface need not be on during idle periods thereby saving power. The IEEE 802.11b specification [15] includes schemes by which access points buffer packets so clients can sleep for short intervals. Our Power-Save-Mode for switch ports is similar. We augment this idea by incorporating a dynamic and automatic sleep period to maintain latency bounds.

Wake-on-LAN [7] is an Ethernet networking technology that allows a computer to be woken up by an incoming network packet. This is useful allowing computers to be put to sleep during idle periods leaving the Wake-on-LAN enabled network interface card switched on. The computer can be woken up remotely using a special "magic" packet. The Wake-on-Packet facility that we assume in our algorithms and advocate for future switch designs is similar to Wake-on-LAN. Using this facility, switch ports that are powered down are automatically woken up on any incoming packet. Other recently reported research have assumed similar support [22, 20].

Gupta et al. [17] was an early work that identified the Internet's high power consumption and explored performance effects due to network-wide coordinated and uncoordinated sleeping. They followed their work in [20] by devising low-power modes for switches in a campus LAN environment. While our Time Window Prediction scheme is related to that of [20], we take better advantage of extended idle periods as our scheme is based on observations in a time-window while [20] requires the port to be on throughout the idle period. This advantage is significant when the traffic patterns are bursty with long idle periods. Also, we introduce latency bounds and investigate the effect on latency and packet loss.

Gunaratne et al. [13], Energy Efficient Ethernet [3] and Nedevschi et al. [22] look at intelligent scaling of switch link speeds depending on network flow. For example, a link need not be active at 1 Gbps if 100 Mbps is sufficient for the traffic. An important practical problem is that the speeds on the switch are discreet (10 Gbps, 1 Gbps, 100 Mbps and 10 Mbps) and hence taking advantage of this automatic scaling would require vast differences in the traffic flows. This problem is acknowledged in [13] where they test their algorithm with non-existing link speeds of 20 Mbps, 30 Mbps and 40 Mbps. [22] also admits to this where the testing on the exponential distribution of speeds gives relatively worse results. Automatic scaling of link speeds also incurs the overhead of auto-negotiation of link speeds between the endpoints. Note that our Time Window Prediction and Power Save Mode schemes can be co-deployed with the automatic link speed scaling algorithm.

Nedevschi et al. [22] also talk about a "buffer-and-burst" (B&B) scheme where the edge routers shape the traffic into small bursts and transmit packet in bunches so that the routers in the network can sleep. This scheme appears to have the following drawbacks: (1) It is not effective for outgoing traffic that originates from the internal nodes and goes outside (e.g., Web server response to external requests), and (2) It requires all the edge routers to adopt the scheme to achieve power reductions. In contrast, our scheme looks at traffic in both directions and is suited to incremental deployment since it is stand-alone and does not require a network-wide coordination. This dramatically reduces the barrier to adoption of our proposal.

Gunaratne et al. [13] addresses the problem of reducing power consumption of desktop PCs using protocol proxying on the NICs or at the first-level switches. We believe this is complementary to our work on reducing the power consumption of switches.

Dynamic Ethernet Link Shutdown [18, 19] talks about an Ethernet level protocol that dynamically shuts down the link by predicting inactivity using packet inter-arrival times.

8 Conclusion

We proposed three power reduction schemes that focus on opportunistic sleeping and lightweight alternatives during idle or low-activity periods. We have also shown a novel architecture for buffering ingress packets when the port is powered down. The results of our simple power reduction schemes on real-world traces in terms of power savings as well as minimal performance degradation are encouraging. We also explore the power savings in the presence of smart hardware features like Wake-on-Packet and fast transitioning of power states in the switch ports, and the significant advantage offered by these leads us to recommend such features in future switch designs.

References

[1] Cisco Catalog. *http://cisco.com/en/US/prod/collateral/switches/ps5718/ps708/product_data_sheet0900aecd8017376e.html*.

[2] Cisco Power Calculator. *http://tools.cisco.com/cpc/launch.jsp*.

[3] Energy Efficient Ethernet. *http://www.ieee802.org/802tutorials/july07/IEEE-tutorial-energy-efficient-ethernet.pdf*.

[4] IEEE P802.3az Energy Efficient Ethernet Task Force. *http://www.ieee802.org/3/az/index.html*.

[5] Ipmon Sprint. The Applied Research Group. http://ipmon.sprint.com.

[6] SPEC Benchmarks. *http://www.spec.org/*.

[7] AMD Magic Packet Technology, 2004. *http://www.amd.com/us-en/ConnectivitySolutions/TechnicalResources/0,,50_2334_2481,00.html*.

[8] Energy Efficient Ethernet, Outstanding Questions, 2007.

[9] J. Flinn and M. Satyanarayanan. Energy-aware adaptation for mobile applications. In *17th ACM Symposium on Operating Systems Principles*, Dec 1999.

[10] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat and R. P. Doyle. Managing Energy and Server Resources in Hosting Centers. In *Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles (SOSP 01)*, Oct 2001.

[11] Yuvraj Agarwal, Ranveer Chandra, Alec Wolman, Victor Bahl, Kevin Chin and Rajesh Gupta. Wireless wakeups revisited: Energy management for voip over wi-fi smartphones. In *5th International Conference on Mobile Systems, Applications, and Services (MobiSys '07)*, Jun 2007.

[12] Amit Jardosh, Gianluca Iannaccone, Konstantina Papagiannaki and Bapi Vinnakota. Towards an energy-star wlan infrastructure. In *The 8th IEEE Workshop on Mobile Computing Systems and Applications (HOTMOBILE 2007)*, Feb 2007.

[13] C. Gunaratne, K. Christensen and B. Nordman. Managing Energy Consumption Costs in Desktop PCs and LAN Switches with Proxying, Split TCP Connections, and Scaling of Link Speed. In *International Journal of Network Management*, pages 297–310, 2005.

[14] Eugene Shih and Paramvir Bahl and Mike Sinclair. Wake On Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices. In *8th annual international conference on Mobile computing and networking (Mobicom 2002)*, Sep 2002.

[15] IEEE802.11b/D3.0. Wireless lan medium access control (mac) and physical (phy) layer specification: High speed physical layer extensions in the 2.4 ghz band. 1999.

[16] K. W. Roth, F. Goldstein and J. Kleinman. Energy consumption by office and telecommunications equipment in commercial buildings - volume i: Energy consumption baseline. In *Tech. Rep. 72895-00, Arthur D. Little, Inc*, Jan 2002.

[17] M. Gupta and S. Singh. The Greening of the Internet. In *ACM SIGCOMM*, Aug 2003.

[18] M. Gupta and S. Singh. Dynamic Ethernet Link Shutdown for Energy Conservation on Ethernet Links. In *IEEE International Conference on Communications*, june 2007.

[19] M. Gupta and S. Singh. Using Low-power Modes for Energy Conservation in Ethernet LANs. In *IEEE INFOCOM (Minisymposium)*, May 2007.

[20] M.. Gupta, S. Grover and S. Singh. A Feasibility Study for Power Management in LAN Switches. In *12th IEEE International Conference on Network Protocols*, Oct 2004.

[21] Martin Arlitt and Tai Jin. Workload characterization of the 1998 world cup web site. In *Technical Report HPL-1999-35R1, HP Laboratories,*, Sep 1999.

[22] Sergiu Nedevschi, Lucian Popa, Gianluca Iannaccone, Sylvia Ratnasamy and David Wetherall. Reducing network energy consumption via sleeping and rate-adaptation. In *5th USENIX Symposium on Networked Systems Design and Implementation (NSDI '08)*, Apr 2008.

[23] Srikanth Kandula, Dina Katabi, Bruce Davie and Anna Charny. Walking the tightrope: Responsive yet stable traffic engineering. In *ACM SIGCOMM*, Aug 2005.

[24] W. Yuan and K. Nahrstedt. Energy-efficient soft realtime cpu scheduling for mobile multimedia systems. In *19th ACM Symposium on Operating Systems Principles*, Oct 2003.

[25] W-D. Weber X. Fan and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *The 34th International Symposium on Computer Architecture (ISCA 2007)*, August 2007.