eBond: Energy Saving in Heterogeneous R.A.I.N

Marcus Hähnel, Björn Döbel, Marcus Völp and Hermann Härtig
{mhaehnel,doebel,voelp,haertig}@tudos.org
Technische Universität Dresden
Faculty of Computer Science
Institute of Systems Architecture
Operating Systems Group

ABSTRACT

Network energy is a significant, although not the largest, cost factor in medium to large scale server installations. On the other hand, most server installations work with redundant link and infrastructure layouts to reduce the risk of network outages. Introducing eBond, an energy-aware bonding network device, we exploit possible heterogeneities in these redundant layouts to adapt network device energy consumption to dynamic server bandwidth demands. Replaying the trace of a realistic scenario in a simulation of eBond with fine grain energy profiles measured at two network cards we achieve energy savings up to 75 % for the server-side network interconnect.

Categories and Subject Descriptors

Software and its engineering [**Operating systems**]: Power Management; Networks [**Network protocols**]: Network layer protocols

Keywords

energy; network; server; eBond; network card, bonding

1. INTRODUCTION

Energy demand is one of the larger cost drivers in large scale server installations. Modern data centers consume between 10 % and 15 % of their total operation power in network links and infrastructure [12]. This demand translates into a significant though not the highest cost factor on the power bill.

In this paper, we focus on optimizing network link energy in medium to large scale server settings by adjusting the power demand of server-side network cards to the actual bandwidth requirements of the servers. Our approach is based on the observation that network links are typically redundant to limit the risk of network outages. Rather than connecting servers with the same high-end network interface cards (NICs), for example two 10 gigabit Ethernet (10 GbE) NICs, we propose to introduce heterogeneity by also including more lightweight connections such as Gigabit Ethernet (GbE) NICs. In the rare case of failure of all high bandwidth connections, these inexpensive cards may still offer some limited bandwidth to an otherwise disconnected server. However for the more common case of a medium loaded data center, being able to scale energy consumption by switching between energy-demanding high bandwidth cards and low power connections gives room for significant energy and cost savings.

Our proposed setup is especially beneficial for installations with a high difference of demand during day and night times or other cyclic demand variations. These variations can often not be compensated by load balancing because latency requirements prohibit relocating the system load to other global regions. Prime examples include on-line gaming services such as OnLive [17] or Google's live search.

After providing the necessary background and relating our work to the works of others, Section 3 presents the setup and results of our study of two wired network cards. Although energy efficiency has been a hot topic for quite some time now, we found that recent network interface cards still offer only limited power scaling possibilities and that switching to a lower bandwidth card leaves room for power savings.

Motivated by these results and realizing that server resilience demands for alternate connections anyway, we developed eBond — an energy-aware bonding network device — which we introduce in greater detail in Section 4. eBond exploits the possibility to layout network infrastructure heterogeneously and builds on channel bonding, which sometimes is also called redundant array of inexpensive networks (or R.A.I.N) [6], to reroute traffic to low power infrastructure if the current server load tolerates the reduced bandwidth of this infrastructure. To evaluate the performance of eBond, we have implemented a network power simulator (see Section 5) to replay exactly the same network traces for different NIC characteristics. Section 6 presents the results of our evaluation using traces of two real-world scenarios. We show, that we can save up to 75 % of the energy used by the network cards when using our approach.

eBond integrates itself into our larger vision of energy-adaptive computing and networking. In Section 7 we conclude this paper highlighting our vision of energy-adaptive computing.

e-Energy'13, May 21–24, 2013, Berkeley, California, USA.

2. BACKGROUND & RELATED WORK

Channel bonding was first introduced in 2000 by the IEEE 802.3 group [8] and has since been used to improve outage resilience and network bandwidth. It does so by coupling redundant links into one virtual link [7]. In our setup, we will use transmit load balancing (mode 5) to redirect traffic between the 10 GbE and the GbE NIC. On the server side, eBond switches between these two cards depending on the amount of outgoing and incoming traffic. No special support is required by the cards or the switch with the exception that one of the two NICs has to be able to take over the MAC address of the respective other. In case eBond decides to power down one of the two NICs, the other card will take over and respond to all traffic sent to this MAC. While Imaizumi et al. [13] recognize the potential for energy savings in link aggregated setups, they do not extend this to heterogeneous device configurations. To the best of our knowledge, channel bonding has not been used before for server-level network link energy optimizations in heterogeneous setups.

Research on energy optimization typically focuses on the CPU [19, 2] or uses whole system measurements [18] to characterize a system's current demand. In these latter works, the power consumption of individual devices is often difficult to isolate, in particular as these devices are still lacking the power measurement equipment that was recently introduced in CPUs [14, 11].

In 2010 the IEEE ratified the IEEE 802.3az standard that promises energy-efficient Ethernet [3]. The approach taken there is to power down a port that is not used, providing a "sleep mode" for the Ethernet port. At the same time the device is never considered off-line, as a low power connection to the other side is kept alive and refreshes the sleep status, or wakes the port if required. While this can yield important improvements in energy consumption it is mainly beneficial for network connections that are idle for longer periods of time — a setting not always present in highly loaded web services.

An alternate approach was suggested by Gunaratne et al., trying to adapt the link rate depending on demand [9]. While this is an interesting approach it requires specialized hardware support and especially requires the processing units on the chip to adapt sufficiently depending on the link rate to reach relevant energy savings. We found that such a scaling was not possible with our cards.

Wireless network energy has been studied extensively in the setting of mobile devices [1]. The resulting models, which weigh throughput and power consumption with other factors prevalent in mobile settings such as battery lifetime, tend to become very complex [5].

Sohan et al.'s study on 10 GbE NIC energy consumption [20] confirms our findings that NIC power often does not scale well with bandwidth. They conclude that further hardware improvements are required to make the network energy scale. Our solution is entirely based on software assuming heterogeneity in redundant links, which server installations have to provide anyway. This provides a convenient intermediate solution while waiting for more efficient hardware designs to be able to scale power near-proportionally to bandwidth.

Of course, network link energy is only a part of the energy spent for data center networking. Heller et al.[12] focus on optimizing the infrastructure's energy costs by rerouting links to turn off unused switches and Gupta et al. have analyzed the feasibility of power management in those devices [10]. Our work is orthogonal to these results and may allow for additional savings.

3. A STUDY OF TWO WIRED NETWORK CARDS

Sohan et al.'s study [20] on network energy consumption provides power values for a wide range of cards. However, we needed higher resolution results with many measurement points to demonstrate the power savings of our setup with a reasonable high accuracy. To obtain these detailed energy models, we measured the power consumption of two exemplary network interface cards at varying bandwidths.

3.1 Methodology

To obtain the energy profiles, we created a direct private network link between two Intel Core-i5 PCs (i.e., no switches, etc.). The machines were connected using a single, 3 m long CAT6 network cable. To get precise, high-resolution power consumption measurements, we installed a riser card and cut the 3.3 V and the 12 V rails of the ribbon cable. Into this riser card we then plugged the to be measured network card. We employ a Yokogawa WT-210 [22] digital power meter, that is capable of measuring current and voltage at the same time with a sampling frequency of up to 10 Hz. Amperage was measured by routing all 3.3 V and 12 V rails through one dedicated Yokogawa power meter for each of the two voltage levels. The power meter provides integrated shunts, which are necessary for measuring currents. The voltage was taken between the riser card's corresponding voltage rail and one of the ground wires of the system's power supply using the voltage inputs of the Yokogawa power meters. This setup ensures the highest precision because also variations in voltage are recognized and factored into the total power consumption.

We measured the power consumption of an 1 Gbit Intel EXPI9301CTBLK network card with an E25869 (B) on card CPU as well as a 10 Gbit Intel Ethernet Server Adapter X520-T featuring an E76983 (A) CPU. The cards have a manufacturer claimed typical power rating of 1.9 W [16] and 18 W [15], respectively.

3.2 Challenges

Our first attempt to obtain power characteristics of our network cards was to run a microbenchmark, which gradually increased the bandwidth in steps of 1 Mbit/s after every measurement interval. At a first glance, this benchmark produced a seemingly nice profile and was reproducible with nearly identical results over several independent runs. However, it turned out that after we degraded the bandwidth again at the end of one run to obtain further results, the power consumption for this degraded bandwidth did not match the original power consumed when running our microbenchmark at this bandwidth. More important however, the power demand for this bandwidth did not adjust itself over time but continued to deviate while we increased our power sampling times. We are not absolutely certain what caused these deviations, but assume that this is due to some chip internal logic that adjusts parameters based on a history of previous usage. To compensate for this effect, we repeated our benchmark switching randomly (with a uniform distribution) between the to be measured bandwidth settings.

3.3 Results

Figure 1 shows the results of one measurement run. Each bandwidth level was held for 10 seconds. The figure shows, that the measured power did not deviate during a single interval but was also not constant at the same bandwidth in different intervals. There is no guarantee that a higher bandwidth leads to higher power levels. We experienced this same effect in our previous approach for generating network card profiles.

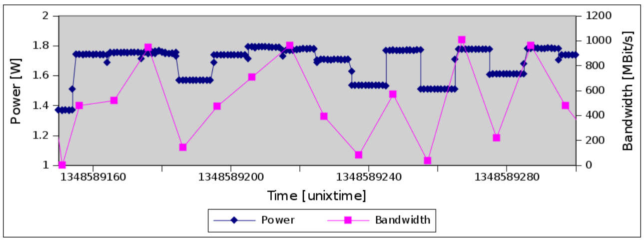


Figure 1: The individual power levels as seen during the microbenchmark using randomly selected bandwidth levels

The random distribution of the trace however, allowed us to gather detailed information, including minimum, maximum and average power consumption levels at the different bandwidths. The results of this extraction are the detailed power profiles of the network interface cards, one of which is shown in Figure 2a. Plotted is the power consumption for varying receive bandwidths. The Figure shows the 1 Gbit/s Ethernet card. Figure 2b shows a comparison between the average send and receive powers for the gigabit card, which shows nearly no difference between the two power levels. Even the dips are nearly the same. Figure 2c zooms into this effect by showing the difference between send and receive power ($P_{send} - P_{receive}$) on the various bandwidth levels. The variation is well below 0.02 W.

For the 10 Gbit/s card, we only show the receive power as the power needed for sending was virtually identical. We further performed our benchmark in steps of 10 Mbit/s intervals to reduce both profile creation and simulation time. A plot with the minimum, maximum and average power consumption calculated over multiple consecutive runs, is plotted in Figure 2d. For each of the displayed bandwidth values there were at least five measurements. In general, the trend of the power levels to increase can be seen with both cards, but the 10 Gbit/s device scales very poorly with load (please note the offset of the y axis). This adapter is also the only device using the 12 V rail, albeit there were no variations of the power levels on this rail when the bandwidth was changed.

One aspect not shown in the above Figures is when the cards are sending and receiving at the same time. For the 10 GbE device this case is nearly indistinguishable from the sending/receiving curves. There is nearly no increase in power when sending and receiving at a bandwidth, compared to only doing one of these operations. The Gigabit Ethernet adapter shows different characteristics between only sending, only receiving or performing both operations at the same time. In order to visualize this difference, Figure 3 illustrates the whole profile as a breakdown of send and receive bandwidth. We use the same breakdown in our simulator. The axes show the respective send and receive bandwidths. The colors indicate the power consumption at those bandwidth levels.

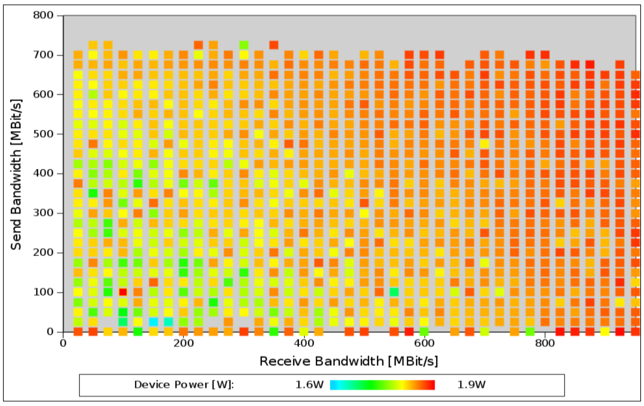


Figure 3: The individual power for different send/receive-bandwidth combinations, as determined by the microbenchmark

3.4 The Odd One Out

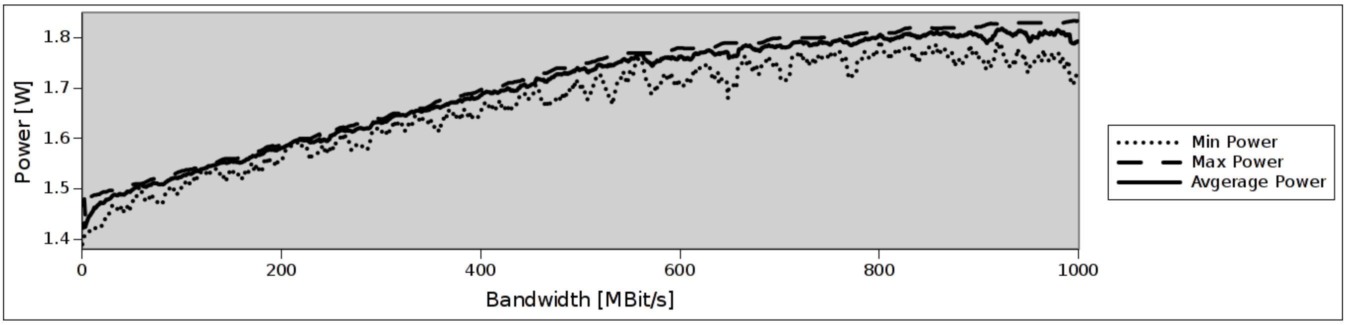
During our benchmarks we also tested one further network card: a gigabit Ethernet card manufactured by Intel (model number EXPI9300PTLPBLK). It belongs to the PT family of Intel gigabit network cards and is rated with a typical power consumption of 3.3 W. We found, that this card does not scale with bandwidth at all. If the interface is up it consumes a near constant 1.82 W increased only to 1.83 W when we draw the full bandwidth from the card. Curiously, it is also the only card we have seen to conserve energy when the interface is powered down without unplugging the cable. While the CT series card can achieve even lower power levels when the cable is unplugged, this procedure is infeasible in a data center environment. Power-saving methods must be controllable by software to be automated or at least remote controlled. Table 1 shows an overview of the relevant measurements of the three different cards in the most significant situations.

3.5 Summary

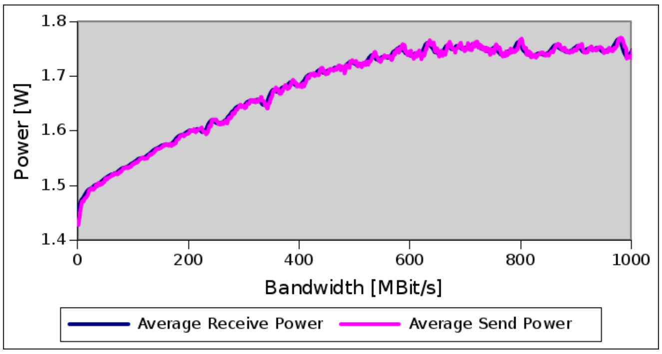
The results of our investigation goes in line with the measurements performed by Sohan et al. in their study on 10 GbE NIC energy consumption [20]. We found, that network cards still do not scale their power consumption with bandwidth requirements, at least not in a way that is comparable, in terms of saved power, with using dedicated lower power network cards.

Moreover, we found that the power characteristics of network cards vary widely even within similar cards from the same manufacturer. We have created a profile for a 10 GbE and a Gigabit Ethernet card, which characterizes the power requirements of these cards at different bandwidth levels. We also analyzed the capability of these network cards to switch themselves into lower power modes when the cable is unplugged or when the interface is disabled.

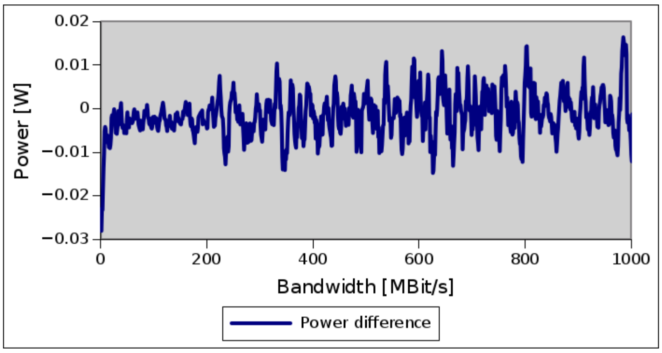
We were only able to evaluate add-on network cards, because the correct instrumentation of a mainboard's components is very hard. While we are currently investigating such



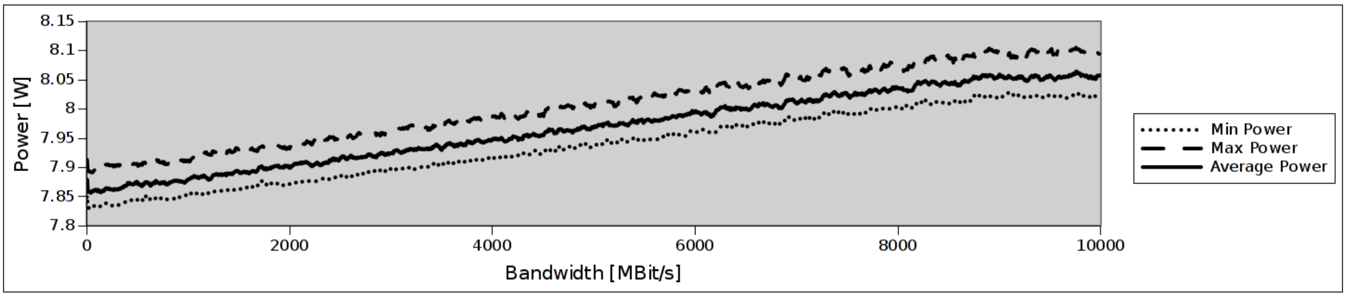
(a) Gigabit ethernet, receiving



(b) Gigabit ethernet, average sending / receiving



(c) Gigabit ethernet, average sending / receiving



(d) 10 Gigabit ethernet, receiving

Figure 2: Minimum, maximum and average power consumption of the two measured network cards with randomly selected bandwidth settings.

a setup, it was not available in time for the publication. We expect, however, that such measurements would not change the general message of the measurements. The processing capabilities of server grade on-board network cards comparable to add-on chips, and, baring overhead introduced by PCIe, should not be significantly different. Especially, if we combine a possible on-board 10 GbE card with an PCIe Gigabit Ethernet card, we still expect there to be a difference in energy consumption that favours the Gigabit card, albeit at lower savings.

4. EBOND: ENERGY-EFFICIENT BONDING

Based on the observations made in the previous section, we propose eBond, an energy aware network scheduler for adjusting the servers' network energy consumption to their loads. To do so, eBond exploits features of the Linux bonding interface [6] operation of network cards.

After a short introduction to the concept, we present some scenarios where we think eBond will be beneficial to reduce energy costs while keeping the service at a high quality level. After that, we give a sketch of the algorithm for scheduling the network cards.

4.1 Concept

As many server installations are equipped with redundant network interfaces [4] we propose to employ a heterogeneous scheme, with high energy network cards to handle the expected peak network load and one or more low energy cards as additional connection and backup.

eBond always chooses the more energy efficient cards as long as these cards can sustain the requested bandwidth using channel bonding both for switching between cards and for sustaining bandwidth if the requirements exceed the bandwidth of a single card. The decision which cards to activate is based on the observations we made in the previous

	10 Gigabit X520-T2	Gigabit CT EXPI9301CTBLK	Gigabit PT EXPI9300PTLPBLK
Interface down, cable unplugged	7.35 W	0.08 W	0.7 W
Interface down, cable plugged in	7.88 W	1.35 W	0.7 W
Interface up, no transfer	7.88 W	1.35 W	1.82 W
Interface up, transfer at full duplex bandwidth	8.10 W	1.92 W	1.83 W

Table 1: The measured network cards at various modes

section. In the case of only two cards (one 10 GbE and one 1 GbE), this decision is to pick the GbE card whenever the bandwidth is below one gigabit per second.

Our system is prepared to work directly with the power profiles to also handle scenarios where the decision which card or combination of cards is more power efficient cannot be made as simple.

We now present the scenarios where the usage of the eBond interface is beneficial and then give the details of the eBond systems algorithm. To simplify the following discussion, we restrict ourselves to two card setups. In these setups, the bandwidth threshold sufficed as a card selection criterion. We show that eBond may save 75 % of NIC energy consumption in a realistic scenario and while only introducing a negligible amount of overload.

4.2 Scenarios

Clearly, channel bonding is most beneficial in scenarios where a significant portion of the requests can be handled by the low bandwidth card. These are scenarios with significantly higher peak bandwidth requirements or with large bandwidth variations.

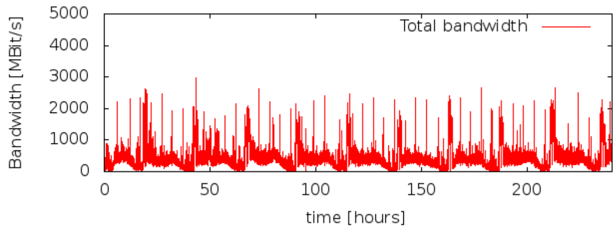
One possible such scenario is that of a server that has regular variations in its load, like a weekly cycle or a day and night cycle. As most servers aim to serve users close to them, a day and night cycle should be observable for a lot of FTP servers or web services with local server infrastructure like Wikipedia or OnLive. Usually such web services are distributed across the globe to reduce access latency and to pay for cheaper, intra-continental traffic. But this also leads to the above mentioned day/night cycles in bandwidth, that can not simply be compensated by re-routing traffic from other continents, without loosing the low-latency property.

While we refer to the variations with the terms "day cycle" and "night cycle" for high and low bandwidth times respectively the concept is of course valid for other variation patterns as well. We will use these terms for the remainder of this paper without loss of generality.

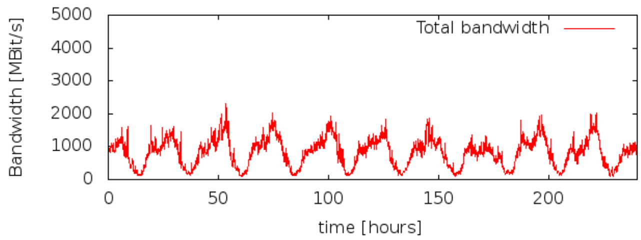
In order to be useful, eBond also requires part of the bandwidth to be in the range of more than one NIC type. It is, for example, not beneficial to have traffic that varies between 2 Gbit/s and 10 Gbit/s, and only have 10 Gbit/s network cards available. In that case the main optimization would be to choose energy efficient 10 Gbit/s network cards. An optimal scenario for eBond would have a night cycle of well below 1 Gbit/s and a day traffic of well over 1 Gbit/s.

Further, we require the small bandwidth network card, which is to serve the night cycle, to have a lower power footprint than the high bandwidth card at at least one bandwidth range. Judging from our analysis in Section 3 and the results presented in [20] by Sohan et al, we claim that this is the case with most modern network equipment.

We do not necessarily require a strict ordering of the network cards' energy efficiency, there might be some overlap, where the high bandwidth card is more energy efficient than the low bandwidth card or that one card only covers the mid bandwidth ranges while the other covers high and low ranges at the same time. But we have not yet encountered such a setup in practice.



(a) Bandwidth requirement of a Debian/Ubuntu FTP server over 10 days



(b) Bandwidth of the uplink of a dormitory complex over 10 days

Figure 4: Bandwidth of example scenarios

For our later analysis we chose two scenarios that fit the criteria discussed above:

1. *Debian/Ubuntu FTP* represents a trace of our local Debian/Ubuntu mirror over 43 days. The trace recorded the bandwidth every 5 seconds, separating incoming and outgoing traffic. A plot of the total traffic is shown in Figure 4a.

2. *Uplink of a dormitory complex* captured a trace of up- and downstream bandwidth and was stored in rrd format. Because of this, the data is available in resolutions of 1, 5, 30 and 360 minutes for the most recent 2, 10, 60 and 720 days respectively. We used the data with 5 minute resolution as shown in Figure 4b for our experiments, because it represents the best trade-off between resolution and simulated time.

4.3 Implementation and Design

The initial version of eBond runs the algorithm shown in Algorithm 1.

Algorithm 1 Basic eBond algorithm

$cur_{card} \leftarrow default_{card}$

while `True` **do**
  $P_{min} \leftarrow \infty$
  $opt_{card} \leftarrow$None

  $BW_{send} \leftarrow (sentBytes(t) - sendBytes(t-i))/i$
  $BW_{recv} \leftarrow (recvBytes(t) - recvBytes(t-i))/i$

  $\triangleright$ Find the optimal card for the bandwidth
  for all $card \in Cards$ **do**
    $P \leftarrow card.getPower(BW_{send}, BW_{recv})$
    if $P < P_{min}$ **then**
      $P_{min} \leftarrow P$
      $opt_{cart} \leftarrow card$
    end if
  end for

  $opt_{card}.activate()$
  wait until card **is ready**
  $cur_{card}.powerDown()$
  $cur_{card} \leftarrow opt_{card}$
  sleep i
end while

In each iteration the up- and downstream bandwidth requirements are determined by querying the statistics of the bonding interface. The bandwidths are always calculated over a user defined sliding window. While the above algorithm assumes the window to be equal to the reconfiguration interval, it is also possible to use other window sizes.

Then for each available network card in the bonding interface the power levels at the required bandwidth levels are calculated and the minimum power network card is identified. The such selected card is activated and the previous card deactivated. After that, the eBond driver sleeps until the next reconfiguration interval.

The reconfiguration interval gives the minimum time between checks of the bandwidth. If sudden surges occur outside this timeframe, eBond may not be able to cope with these effects. This may lead to service level agreement (SLA) violations, as a low power network card may not be able to cope with the increased traffic.

Using this basic algorithm, we observed a flutter effect leading to either increased power consumption or an increase in SLA violations or both, depending on the workload. This happens when traffic is oscillating around the trip point between two cards. In this case, there may be a lot of changes between the active cards, leading to an actual increase in total power consumption, because both have to be driven at the same time for a short amount of time to ensure that no connections are lost. We introduced two methods to mitigate this effect.

First we introduce a hysteresis: We only switch to the lower power interface, if an increase in bandwidth by a factor we term *predictor* will still result in the new card being more energy efficient. An unsupported bandwidth has infinite power consumption in the card, in order to avoid the card being chosen.

We set this predictor to 0.1 in some of our experiments to simulate a 10 % increased bandwidth requirement.

Still this does not fix repeated sudden high spikes in the network load. We address this issue by introducing a cool down time, in which the interface may not be switched down to a lower bandwidth card. This is based on the observation, that spikes often have a temporal locality. During this cool down time we only allow switching to a higher bandwidth interface.

This also limits the switching frequency and thus the time that two network cards are turned on at the same time.

Please be reminded, that this cool down is *not* equivalent to increasing the reconfiguration interval as this would prevent the interface from switching to a higher bandwidth when required. The cool down only limits switching down. As such it presents a trade-off between higher power consumption (longer cool down) and higher number of SLA violations (shorter cool down).

4.4 Advantages of the Bonding Interface

We chose the bonding interface for our work, because it presents an easy implementation of energy aware network card selection. Not only is bonding a technique that is available in most data center switch technology but it also allows us to not care about modifications of the routing tables or similar intricacies of the network stack, that appear when switching between network interfaces [1].

Also bonding is mostly already done in data centers, where availability is an important service selling point. Our proposal only improves on the existing bonding techniques by suggesting the usage of heterogeneous network cards as backup links. We exploit this heterogeneity as a potential for energy improvements. The price of our approach is a slightly increased mean time to recovery in the event of a link failure because the other card needs to be turned on first.

To disclose all risks we must also mention that our approach can lead to reduced available bandwidth should all the available high bandwidth links fail. This could be mitigated by providing more high bandwidth links, that could be powered down during normal operation.

This setup gives the operator also an easy choice to drive a single server in either traditional bonding modes or operate in energy efficiency mode using the eBond system.

The bonding interface further allows us to extend the system to any number of cards, and not be limited to two cards. It is even possible to use completely different cards and physical layers, as long as their drivers and switch technology support bonding.

5. SIMULATOR

We decided to use a simulator to evaluate the effectiveness of the eBond interface. There are several reasons behind this decision. The first is simulation time. We wanted to use real world profiles as were introduced in Section 4.2, which span several weeks of sever operation. This long time span is required, because we need to capture several day an night cycles, spikes and irregularities, to demonstrate the effectiveness of our eBond interface across a wide range of scenarios. To run these long term test cases with various parameters would take years. This is clearly unacceptable.

Shorter periods either do not capture the effects we need to demonstrate the effectiveness of eBond, biasing the result in one or another direction, or are synthetic, making the evaluation less realistic compared to the load scenarios of real world servers.

The second reason is, that we did not have enough high precision measurement technology to capture two cards simultaneously on all power rails with sufficiently high accuracy.

We believe that our above shown method for capturing network card energy behavior presents us with sufficiently precise profiles for a simulation to capture the energy consumption of the network interfaces with high enough detail, and only limited error. This can also be seen in the amount of variation we saw in our profiles as detailed in Figures 2b-2d in Section 3.3.

The simulator is implemented as a python script, which evaluates the power consumption of the network cards based on the profiles taken as described in Section 3. The simulation interval is based on the data point interval of the scenarios' datasets, but never smaller than the eBond reconfiguration interval. The eBond algorithm, including cool down and predictor, is used to determine active network cards.

The configuration file for the simulator is the same that we also used for the eBond interface. No further settings are required. The same is valid for the energy profiles of the network card. This keeps the configuration overhead minimal and ensures consistency between the real eBond interface and the network simulator.

The simulator then evaluates the data and generates an energy profile that we show for our two demo scenarios in Section 6 as well as detailed statistics on SLA violations and network card usage.

The sources of the simulator, together with the NIC energy profiles, will be made available in time for the conference at our github repository [21].

6. EVALUATION

To determine the prospective energy savings of eBond we used the two scenarios introduced in Section 4.2.

We replayed 43 day traces of a Debian/Ubuntu FTP mirror and 10 day traces of a Dormitory network uplink in our network simulator using different network card scheduling policies. The simulator accumulates the network bandwidth used by the network cards during transfer and idle times. This power is also recorded in a trace, by matching each bandwidth adjustment against the power profiles presented in Section 3. We first present the detailed results of the FTP scenario and then provide a short summary of the results of the Dormitory uplink scenario.

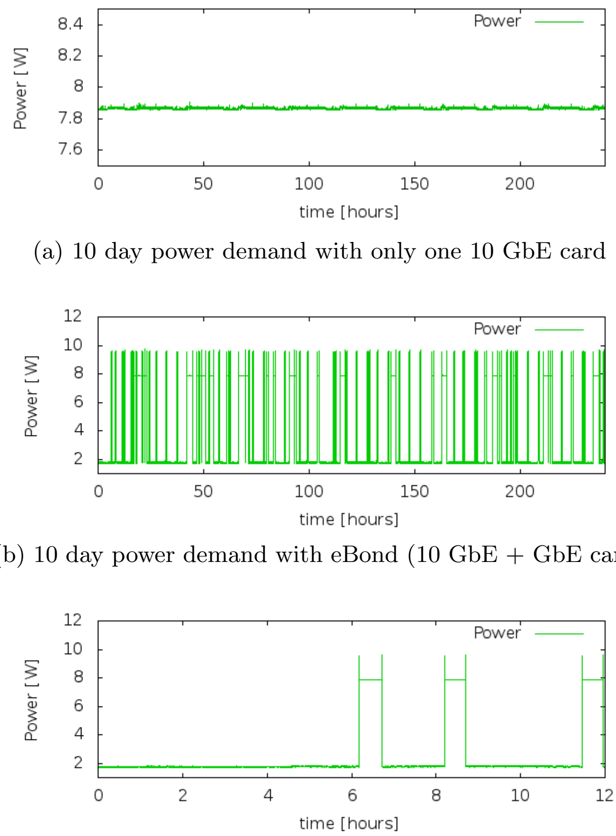
6.1 Detailed FTP Scenario

The graphs in Figure 5 show energy characteristics of the FTP trace over a 10 day period in different scenarios. Figure 5a presents the average power consumed in the traditional setting where all load is served by a single 10 GbE card. When we compare this consumption to the bandwidth graph seen in Figure 4a we can see a clear optimization potential.

In Figure 5b, we present the power consumption graph of the two network links when combined into an energy-aware bonding device. We see that most of the time, the GbE card suffices to meet the bandwidth demand of the FTP server, yielding a lower average power consumption. Whenever the FTP server's demand exceeds the capability of the GbE card, eBond switches to the 10 GbE card. Figure 5c is a zoomed in view of the first 12 hours of the graph.

Table 2 presents some statistics for the simulation, which confirm these results. We were able to save an average amount of 140 Wh or 74.7 % per day when compared to the single 10 GbE scenario's power demand. The scenario *eBond 3* presents the most aggressive power saving scheme, with no hysteresis or cool-down time, and no load prediction. While this has the most savings because it immediately switches to the most energy efficient card for the current load, it also induces a large number of service level agreement (SLA) violations. These happen, when a requested bandwidth could not be served by the current network card, which leads to lower bandwidth or increased latency from the view of the client.

The setups 1 and 2 present more reasonable configurations that balance energy savings against the number of SLA violations. The concrete parameters that deliver a balanced



(a) 10 day power demand with only one 10 GbE card

(b) 10 day power demand with eBond (10 GbE + GbE card)

(c) 12 hour power demand with eBond (10 GbE + GbE card)

Figure 5: Figures showing the power demand of the network cards in the system for the FTP server scenario. Figure 5a is the setup without the eBond system, while Figures 5b and 5c show power usage with the eBond heterogeneous network channel bonding in the balanced configuration. Figure 5c shows a zoomed in section of Figure 5b.

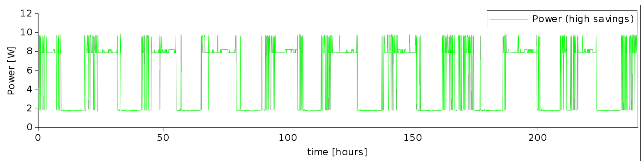
	Single 10 GbE	eBond 1: high savings	eBond 2: balanced	eBond 3: aggressive
Simulated time	43 days	43 days	43 days	43 days
Prediction	-	10 %	10 %	0 %
Cool-down time	-	0 h	0.5 h	0 h
Total energy	8113 Wh	2055.8 Wh	2758.4 Wh	2033.8 Wh
time on 10 GbE	100 %	3.825 %	15.07 %	3.39 %
time on GbE	0 %	96.253 %	84.95 %	96.74 %
SLA violations	0 (0 s, 0 %)	195 (1035 s, 0.028 %)	103 (519 s, 0.014 %)	252 (1265 s, 0.034 %)
Saved energy	0 %	74.7 %	66 %	74.9 %

Table 2: Statistics of the Simulation for the Debian/Ubuntu FTP Server scenario. The SLA violations are given as the number of times the required network bandwidth could not be provided. In parentheses is the total time during which the bandwidth was lower than required together with the percentage of the total time this amounts to.

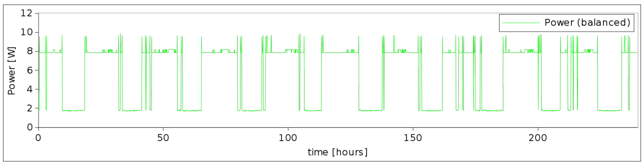
setup heavily depend on the load type and pattern and must be configured specific to the expected server workload.

These savings are already with two quite efficient cards. When considering the results of Sohan et al. [20] there may be even more potential for energy savings in existing server setups.

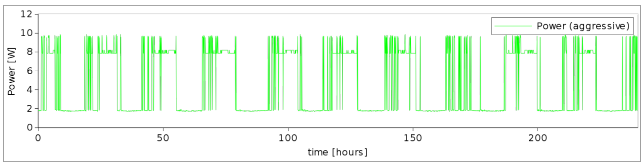
6.2 Uplink scenario



(a) 10 day power demand with high *savings* eBond profile



(b) 10 day power demand with *balanced* eBond profile



(c) 10 day power demand with *aggressive* eBond profile

Figure 6: Figures showing the power demand of the network cards in the system for the dormitory uplink scenario for the different eBond scheduling profiles as shown in Table 2

This second scenario is more stable with less spikes in the bandwidth as was shown in Figure 4b in Section 4.2. On the one hand, this makes the predicting the traffic easier and thereby causes a reduction of SLA violations compared to the FTP scenario. On the other hand, the scenario has also less potential for energy savings, as the required bandwidth does only drop to less than 1 Gbit/s during night time. We ran the same simulation as for the previous scenario using a 30 minute cooldown and 10 % prediction. The result was an energy graph as presented in Figure 6, with the subfigures showing the different scheduling profiles of the network eBond network card scheduler.

The number of SLA violations has been greatly reduced due to the more predictable nature of the network usage compared to the FTP scenario. A comparison of SLA violation times expressed as percentages of the runtime are presented in Table 4.

	high savings	balanced	aggressive
FTP	74.7 %	66 %	74.9 %
Uplink	35.9 %	30 %	43.2 %

Table 3: Energy savings compared to the single NIC setup for the two scenarios under the 3 eBond policies as seen in Table 2

	high savings	balanced	aggressive
FTP	0.028 %	0.014 %	0.034 %
Uplink	0.07 %	0 %	0.14 %

Table 4: Percent of time, that the bandwidth requirement could NOT be satisfied (SLA Violations)

7. CONCLUSIONS AND FUTURE WORK

In this paper, we presented eBond — an energy-aware network bonding interface — to adjust network link energy to the current bandwidth demands of servers in medium to large scale data centers. Our approach exploits heterogeneity in the redundant layout of connections by switching between low power but also low bandwidth network interface cards and high bandwidth cards, which we found to be more demanding and less adaptive. No special infrastructure is required beside the redundant link layout that resilient server installations have to provide anyway. Our simulation of eBond with real-world network traces indicates power savings of up to 75 %. While the power savings depend on the concrete server load scenario our implementation allows for different, user configurable, profiles to select the network card scheduling behavior best fitted for the typical load situation of the network.

There are multiple directions we aim to investigate for future work. On the hardware side, more adaptive network cards, possibly integrating the low bandwidth circuitry next to the high bandwidth setup for better scalability are imaginable with an off-loaded eBond instance to select between. An integration of even more link types such as optical or wireless board-to-board interconnects would be highly interesting as well as other scenarios besides networking.

Further we plan to extend our research to whole heterogeneous network hierarchies, where we include switching technologies into our observations and use different bandwidth switches according to the demand of the attached subnetworks. This will allow us to venture even farther into the domain of whole-datacenter energy efficiency which we also extend in parallel by our work on QOS-based, energy-aware scheduling of resources on individual nodes of the network.

Acknoledgement

The authors would like to thank Waltenegus Dargie for providing his instrumentation setuo for the measurements. We further extend our thanks to Hannes Weissbach for help with the benchmarks and Adam Lackorzynski, Michael Kluge (ZIH), and Maximilian Marx for providing trace data. This work was partially funded by the German Research Council (DFG) through the Collaborative Research Center CRC 912 "Highly-Adaptive Energy- Efficient Systems" (HAEC), the Special Purpose Program "Dependable Embedded Systems" (SPP 1500) and the cluster of excellence "center for Advancing Electronics Dresden" and by the EU and the state Saxony through the ESF young researcher group "IMData".

8. REFERENCES

[1] Agarwal, Y., Pering, T., Want, R., and Gupta, R. Switchr: Reducing system power consumption in a multi-client, multi-radio environment. In *Proceedings of the 2008 12th IEEE International Symposium on Wearable Computers* (Washington, DC, USA, 2008), ISWC '08, IEEE Computer Society, pp. 99–102.

[2] Bellosa, F. The benefits of event: driven energy accounting in power-sensitive systems. In *Proceedings of the 9th ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system* (New York, NY, USA, 2000), ACM, pp. 37–42.

[3] Christensen, K., Reviriego, P., Nordman, B., Bennett, M., Mostowfi, M., and Maestro, J. Ieee 802.3az: the road to energy efficient ethernet. *Communications Magazine, IEEE 48*, 11 (2010), 50–56.

[4] Cisco. Design considerations for high availability and scalability in blade server environments. White Paper, 2009.

[5] Ferretti, S., Ghini, V., Marzolla, M., and Panzieri, F. Modeling the energy consumption of multi-nic communication mechanisms. In *Proceedings of the 2012 IEEE Online Conference on Green Communications (GreenCom)* (2012), IEEE Computer Society.

[6] Foudriat, E. C., Maly, K., Mukkamala, R., Overstreet, C. M., Mathews, L., and Balay, S. RAIN (redundant array of inexpensive networks): Expanding existing networks to support multitraffic performance. Tech. rep., Norfolk, VA, USA, 1994.

[7] Geoff Thompson. Proposal for parallel path trunking in 802, 1998.

[8] Group, I. . Tutorial on Link Aggregation and Trunking. `http://grouper.ieee.org/groups/802/3/trunk_study/tutorial/index.html`, 1997.

[9] Gunaratne, C., Christensen, K., Nordman, B., and Suen, S. Reducing the energy consumption of ethernet with adaptive link rate (alr). *Computers, IEEE Transactions on 57*, 4 (2008), 448–461.

[10] Gupta, M., Grover, S., and Singh, S. A feasibility study for power management in lan switches. In *Network Protocols, 2004. ICNP 2004. Proceedings of the 12th IEEE International Conference on* (2004), pp. 361–371.

[11] Hähnel, M., Döbel, B., Völp, M., and Härtig, H. Measuring energy consumption for short code paths using rapl. In *Greenmetrics* (London, UK, June 2012), S. Low, J. L. Boudec, C. Rosenberg, and G. Zussman, Eds., ACM Sigmetrics.

[12] Heller, B., Seetharaman, S., Mahadevan, P., Yiakoumis, Y., Sharma, P., Banerjee, S., and McKeown, N. Elastictree: Saving energy in data center networks. In *7th USENIX Symposium on Networked Systems Design and Implementation* (San Jose, CA, USA, April 2010).

[13] Imaizumi, H., Nagata, T., Kunito, G., Yamazaki, K., and Morikawa, H. Power saving mechanism based on simple moving average for 802.3ad link aggregation. In *GLOBECOM Workshops, 2009 IEEE* (30 2009-dec. 4 2009), pp. 1 –6.

[14] Intel Corp. *Intel® 64 and IA-32 Architectures Software Developer Manual: RAPL MSR updates.* 2012, ch. 14, p. 32.

[15] Intel Corp. Intel® Ethernet-Server-Adapter X520-T2. `http://www.intel.de/content/www/de/de/network-adapters/gigabit-network-adapters/ethernet-x520-t2.html`, 2012.

[16] Intel Corp. Intel® Gigabit CT Desktop Adapter Product Brief. `http://www.intel.com/content/www/us/en/network-adapters/gigabit-network-adapters/gigabit-ct-desktop-adapter-brief.html`, 2012.

[17] OnLive. Onlive.com website. `http://www.onlive.com/`, 2012.

[18] Pathak, A., Hu, Y. C., and Zhang, M. Where is the energy spent inside my app?: fine grained energy accounting on smartphones with eprof. In *EuroSys* (2012), P. Felber, F. Bellosa, and H. Bos, Eds., ACM, pp. 29–42.

[19] Snowdon, D. C., Petters, S. M., and Heiser, G. Accurate on-line prediction of processor and memory energy usage under voltage scaling. In *Proceedings of the 7th International Conference on Embedded Software* (Salzburg, Austria, Oct 2007), pp. 84–93.

[20] Sohan, R., Rice, A., Moore, A. W., and Mansley, K. Characterizing 10 Gbps network interface energy consumption. Tech. Rep. UCAM-CL-TR-784, University of Cambridge, Computer Laboratory, July 2010.

[21] TU Dresden Operating Systems Group. GitHub Page. `https://github.com/TUD-OS`.

[22] Yokogawa. WT-210 manua;. `http://c418683.r83.cf2.rackcdn.com/uploaded/bu7604_00e_020_3.pdf`, 2012.