Efficient Power Management Using Out-of-Band Signaling

Tobias Vaegs, Muhammad Hamad Alizai, Jó Ágila Bitsch Link, Klaus Wehrle
Communication and Distributed Systems,
RWTH Aachen University, Germany
{firstname.lastname}@comsys.rwth-aachen.de

***Abstract*—A tremendous amount of energy is wasted today, because computing devices are left running all the time even though they are needed only sporadically. Especially in office environments many devices (e.g., printers) are very rarely turned off, because they need to be available from time to time and because it is inconvenient having to switch them on and off manually. Existing solutions, such as Wake-on-LAN (WoL), provide support for managing the power consumption of the network devices remotely using an *always-on* data channel. However, these solutions are inefficient, because power to the network interface has to be maintained even when the host system is asleep just to ensure remote accessibility.**

We propose a Wireless Sensor Network (WSN) based out-of-band signaling architecture for network interfaces which minimizes the systems' power consumption during the large idle periods when nobody is using them. This is done by separating the data and control channels on the Internet-enabled devices using a low-power out-of-band signaling channel based on battery driven, energy scavenging devices. Unlike existing solutions, which only allow parts of the system to go in sleep modes, our architecture allows the whole system, including the main power supply, to be shut down.

Our initial investigation indicates a significant reduction in energy consumption of devices during idle times compared to the existing in-band signaling mechanisms such as WoL.

I. Introduction

Energy efficient operation of networked devices is pivotal in establishing a sustainable green IT infrastructure at our homes and in our offices. The Internet forms the core of our IT infrastructure that includes Internet-enabled desktops, laptops, printers, servers, storage, and IP phones. Approaches such as Wake-on-LAN (WoL) have been proposed to enable sleep-modes on these systems without compromising the remote accessibility enjoyed by always-on systems.

However, over the years, these approaches have not proven successful for several reasons. For example, they rely on heavy infrastructure or application-level support or manual user action, presenting barriers to deployment and use. Similarly, their operation is inefficient since power to the network interface has to be maintained even when the host system is asleep: The network interface actively waits for incoming (in-band) signals and thus continually draws power from the main power-supply whose design is not optimized for powering such a small subsystem.

We present a sensornet based out-of-band signaling architecture for network interfaces that significantly reduces systems' power consumption during large idle periods when nobody is using them. This is done by physically separating the data and control channels on the Internet-enabled devices by employing a low-power out-of-band signaling channel based on battery driven, energy scavenging devices. Unlike existing solutions, which only allow parts of a system to go in sleep modes, our architecture allows the whole system, including the main power supply, to be shut down. The control channel interface is capable of signaling a wake-up or reboot the system automatically when the need arises.

The utility of such an architecture lies in one key question: Is it possible to construct a highly energy and cost efficient out-of-band signaling mechanism to switch the power state of certain devices? The energy overhead and the deployment cost must be substantially smaller than for WoL based network interfaces, since otherwise one should just directly use in-band signaling as employed by the existing approaches. The advances in sensornets research provide an affirmative answer to this question and thus put us in the privileged position to explore the feasibility of such an architecture even further.

In this position paper we suggest interesting directions and opportunities offered by our solution. The remainder of this paper discusses the design of our approach, presents our prototype, and discusses an initial analysis of potential energy savings. A thorough evaluation of the savings for a real deployment is out of scope of this paper.

II. Out-of-band signaling

Our solution for switching devices according to user's needs bases on out-of-band signaling. The basic idea is to control devices over a Wireless Sensor Network (WSN) as a separate signaling channel.

A. Scenario

Consider an environment where several devices are networked, like in a modern office or household. The connection between the devices, which usually bases on the Internet Protocol (IP), we will call the *data channel*. Technologies used for these connections (like Ethernet of WiFi) are optimized to transfer data at a high rate. Although they can also be used for signaling purposes like it is done with WoL, they were not designed with energy-efficiency in mind.

Instead of adjusting the existing data channel connecting our target devices to work more energy-efficiently we propose a separate *signaling channel*. Since this channel is solely

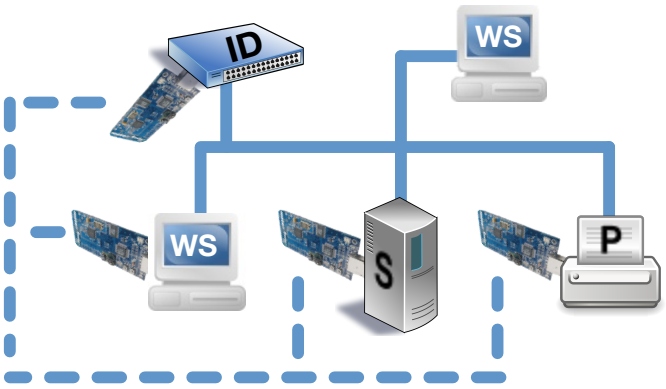

Fig. 1. System design envisioned by us for our prototype: Parallel to the data channel visualized by the solid line we propose a separate signaling channel for controlling purposes using WSN technology visualized by the dashed line.

used for signaling purposes, it does not have to provide a high bandwidth and can be based on energy-efficient WSN technology (i.e., IEEE 802.15.4). Signaling users' needs to the devices in order to control power levels accordingly without having to alter existing communication channels leads to a solution that is cheap and flexible as well as easily and incrementally deployable.

Figure 1 illustrates our design vision. The solid line represents the data channel (i.e., Ethernet and WiFi), which interconnects most of the devices in a given environment, like workstations (WS), servers (S), or printers (P). Usually there is a device or a set of devices acting as interconnecting device(s) (ID) by maintaining the communication between all other devices. This can be everything ranging from a simple WiFi router in a home environment to a very capable server within a company's network backbone.

The ID can send commands over the separate WSN channel represented by the dashed line signaling devices to switch their power state. If a device is capable of IEEE 802.15.4 communication without additional hardware, it may be switched directly over this channel. Devices without this capability can be controlled by an attached sensor node and/or be powered over a power socket controllable over this channel.

The advantage of the ID is that it is usually always running and connected to most other devices over the data channel. So it can be used as a gateway to signal users' needs, which it receives over the data channel, to the connected devices. Additionally to the ID, other devices (like a workstation) could be used to send commands as well so that users can control devices directly from their machines. However, in this case the device being controlled (or the sensor node in control of the corresponding power socket) would have to coordinate commands from several senders.

Since the whole purpose of this design is to save as much energy as possible, and since the sensor nodes are quite limited in their capabilities, it would be better to let them sleep as much as possible. Reacting on received commands would then be their only purpose, and more evolved tasks were handled by more capable devices that are reachable by most other devices over the data channel. Those devices are usually consuming power all the time anyway to perform other tasks and can easily take the additional burden.

The design described here offers a variety of possibilities for controlling the power state of the connected devices. Switching to members of an elaborate set of energy saving states would require the integration of the control logic directly into the devices themselves, as this process is supposed to be very device-specific.

However, with our prototype devices can simply be switched off completely so that they do not consume any energy at all during times in which they are not needed. Yet, they can be switched on again fast and conveniently as soon as they *are* needed. Moreover, our solution is realized with very energy-efficient hardware, so the energy needed to keep the device responsive to the users' needs is reduced significantly compared to currently available solutions directly integrated into the devices (such as WoL).

Although our approach can save energy when used with different classes of devices, consider a networked printer as a use case. The printer is connected to the network over which it receives its print jobs, and additionally it is capable of communicating on the signaling channel over which it can be turned on and off remotely.

The manual case would be that a user who wants to print would have to send a command to switch on the printer before (over the ID) and one to switch it off again after fetching the printouts. This would already save energy, although it does not involve any sophisticated program logic. This is because our solution eliminates the inconvenient need to actually go to the printer to switch it on before printing, which usually prevents users from exploiting the idle times of the device to save energy even though these times might account for a significant portion of the day/week.

Manual switching of the device reflects the users' needs but is still a burden. Detecting users' needs can be performed arbitrarily complex, but in certain scenarios it is very easy. In this use case our prototype software (cf. II-B) running on the ID can switch on the printer when it detects the presence of a corresponding print job and switch it off after some time when no jobs for that printer are issued anymore.

Hence, the printer is switched-off for the majority of its idle time, which is considerably long at nights, over the weekends, and usually even during office hours. This approach promises tremendous power savings as an office printer typically remains powered-on after installation until it is repaired or replaced.

For these scenarios we need the switched devices to also communicate with the ID for example to state when a print job is finished. Another possibility would be to employ appropriate heuristics to determine when a device should be switched on and off, whose derivation lies outside the scope of this paper.

Those heuristics can be based on usage patterns derived over time and should also take the boot-up time of the controlled devices into account.

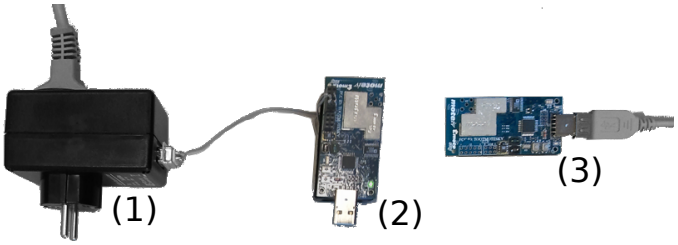

Fig. 2. A photograph of our prototype: One can see the switchable power socket (1) connected to a telosB sensor node (2), that can receive commands for this socket. Moreover, there is a second sensor node (3) connected to a controlling device via USB to send those commands.

B. Prototype

As a proof of concept we created a very simple prototype for our proposed separate WSN signaling channel and the printer use case. We attached a standard telosB sensor node to a switchable power socket with which we are able to switch the printer on and off by enabling/disabling its power supply.

The socket we use[1] provides a D-sub connector as a control interface, that we attached to the exclusive digital I/O expansion connector pins of the sensor node. This way the node can switch the socket and thereby the printer when triggered to do so. A photograph of this prototype can be seen in Figure 2.

The software running on the nodes is intentionally kept very simple to let them work as energy-efficient as possible. We assume one base station node attached to the ID, which sends out the commands for the nodes connected to the power sockets. So the nodes only have to handle the plain command sending and receiving, any other program logic is implemented on the ID itself. Routing simply happens in a tree-based fashion rooted in the base station node to enable multi-hop command relaying.

The software running on the ID monitors the CUPS server in the network. When a printing job appears for a printer, which is known to be switchable, a switch-on command is sent to the sensor node controlling that printer's socket. After a configurable timeout the printer is switched off again, if no more printing jobs exist for it.

Of course, the software can be extended to collect information about usage patterns, room occupancies and the like provided by sensors in the building and nodes attached to workstations and other devices. Decisions on how to switch the device can then be assisted by appropriate heuristics derived from this information.

The sensor nodes connected to a switchable power sockets can employ energy scavenging techniques so that the batteries can be recharged during times when the device which is controlled by the node is running (i.e., when power runs through the switchable socket). The collected energy can then be used by the sensor node to stay responsive during times when the controlled device is completely powered off. Since the node hardware uses only very little energy compared to what the socket provides during times when the device is running, it should usually be possible to keep the node alive, as long as the device is used at least once in a while.

[1]http://www.antrax.de/en/230V-Switchboxes/switch-via-USB-COM-LPT/SwitchBox-Relay

III. Analysis

After presenting our approach we want to give a rough estimate on how much energy can be saved by deploying it.

A. Standby Power

Our prototype helps to reduce the power needed in standby mode, which can be surprisingly much. The survey in [1] gives an overview about how much energy different devices consume during their standby phase, i.e., while not doing anything useful but waiting to be used. Numbers can be found ranging up to several watts with the minimum in the order of one watt, also for computing devices like printers.

This is already significantly bigger than what we need to keep the sensor node responsive, which is all our prototype consumes during devices' idle times.

B. Usage Patterns

Of course, we cannot just compare standby energy consumption of a device we want to control with the energy consumption of a telosB node. We do not consider the power consumption of the base station node, since the device it is attached to (the ID) will be running anyway to perform its other tasks and will not need significantly more energy to power the base station node. Moreover, one base station can be used to control several devices.

However, for a real comparison we need to take into account the device's usage pattern and the additional costs that come with switching the device on and off. For this we use data traces from the Powernet project[2] at Stanford University, where the power consumption of several office devices is monitored constantly during regular usage. We took a look at the data gathered over the whole month June 2011 from three printers, whose power consumption and usage patterns can be seen in Table I.

Three arguments for our approach are directly apparent from these numbers: (1) Most devices are almost never turned off, although they are used only a tiny fraction of the whole time, so they mostly run in standby mode. (2) In this mode the printers consume quite a lot of power. Nevertheless the consumption is realistic considering the values in [1]. (3) Even when the devices are turned off, they consume a significant amount of power.

The total consumption of the three devices over the whole month was 11.38 kWh, 8.31 kWh and 3.64 kWh, respectively.

C. Our Approach

To compute the potential savings of our approach, we calculate its energy costs using the results from [4], where the power consumption of the telosB sensor node was measured and compared to the information in the official data sheet. For

[2]http://powernet.stanford.edu

	avg. consumption [W]			energy spent [%]			time spent regular [%]			time spent OA [%]			total [kWh]		saving
device	off	standby	used	off	standby	used	off	standby	used	off	standby	used	regular	OA	[%]
103	1.39	16.23	413.79	0.01	98.40	1.60	0.01	99.78	0.06	97.19	2.75	0.06	11.38	0.53	95.43
129	1.41	10.59	529.71	~ 0	99.89	0.11	0.01	99.95	~ 0	98.81	1.18	~ 0	7.31	0.12	98.50
151	1.19	6.19	340.68	6.39	87.64	5.97	27.41	72.36	0.09	97.61	2.30	0.09	3.64	0.35	90.66

TABLE I

ENERGY CONSUMPTION DATA FROM THE POWERNET PROJECT [2], [3]. BASIS FOR THESE NUMBERS IS THE DATA GATHERED FOR THREE PRINTERS DURING JUNE 2011. NOTE THAT THE DEVICES CONSUME A SIGNIFICANT AMOUNT OF POWER EVEN WHILE THEY ARE TURNED OFF. WE COMPARE THE TIME AND CONSUMPTION DURING REGULAR USAGE WITH THE TIME AND CONSUMPTION WHEN USING OUR APPROACH (OA).

simplicity reasons we do not consider any duty cycling for now. The goal in the future is to allow the nodes to sleep as much as possible, but their power consumption is already fairly low without any duty cycling. We assume the node to use only 68.4 mW all the time, regardless if the controlled device is running or not. Hence, responsiveness of our system for one device over a complete month (i.e., 24 hours over 30 days) needs 49.25 Wh.

Considering the controlled devices we distinguish three different states: (1) *active* (i.e., the device is turned on and used), (2) *standby* (i.e., it is turned on but not used), and (3) *off* (i.e., its power supply has been cut).

The energy amount for the telosB stated above is spent all the time, because the sensor node has to stay responsive to receive commands. Additionally we have to consider energy costs on top of that for the switchable socket and the device itself. While turned off no additional costs apply, but while the device is in standby or active state (i.e., turned on) the socket needs 1.2 W. Moreover, in the standby state we have to add the device's standby consumption as well as its active consumption during active times.

For our calculations we assume that the device is turned on right before it is used and turned off after its usage has ended and an additional timeout of five minutes has expired. Thus, during this timeout the device would be in the standby state. Whenever the device is not used long enough its power supply is cut setting it to its off state.

The rightmost columns of Table I compare the printers' power consumption as it was recorded (*regular*) as opposed to the consumption calculated by us when using our prototype (*OA*). One can see that at least 90% of the energy can be saved. The savings are less when the devices are used more frequently or when they are turned off manually from time to time, but they are still huge.

D. Boot-up Phases

One more cause of energy consumption we have to consider is the boot-up times of the devices. When a device is running all the time, it constantly consumes some energy. However, when turned off and on in between it saves energy while being off but consumes significantly more power when switching from the off to the on state. We assume the devices controlled by our approach to be idle most of the time, so there will be only very few boot-up phases, but the increase in energy can be very drastic. Unfortunately the Powernet data did not provide this information and since the increase in energy consumption caused by the boot-up process varies significantly depending on the device, it is hard to estimate for the general case.

However, what we can do is calculate how much power a printer would have to spend during boot-up to use up what our prototype would save. The energy saved by our approach without considering boot-up phases is 10.85 kWh, 7.19 kWh, and 3.29 kWh, respectively. During the month of monitoring when using our prototype the printers would have to be switched on by it 227, 95 and 193 times, leading to an available amount of energy per boot-up phase of 47.80 Wh, 75.68 Wh and 17.05 Wh. Even assuming very long boot-up phases of 30 seconds, this would mean that during each of them the printers would have to consume about 5.7 kW, 9 kW and 2 kW, respectively, to defeat the savings gained by our prototype.

ACKNOWLEDGMENTS

The authors like to thank Maria Kazandjieva and her colleagues working on the Stanford Powernet project for kindly providing us with the raw data to analyze as well as Thomas Gerlitz, Norbert Landa and Alexander Roth for helping with implementing our prototype for TinyOS.

REFERENCES

[1] A. Chakraborty and A. Pfaelzer, "An overview of standby power management in electrical and electronic power devices and appliances to improve the overall energy efficiency in creating a green world," *Journal of Renewable and Sustainable Energy*, vol. 3, no. 2, 2011. [Online]. Available: http://link.aip.org/link/?RSE/3/023112/1

[2] M. A. Kazandjieva, B. Heller, P. Levis, and C. Kozyrakis, "Energy dumpster diving," in *Proceedings of the Second Workshop on Power Aware Computing (HotPower)*, vol. 9, October 2009. [Online]. Available: http://sing.stanford.edu/pubs/dumpster.pdf

[3] M. A. Kazandjieva, O. Gnawali, B. Heller, P. Levis, and C. Kozyrakis, "Identifying energy waste through dense power sensing and utilization monitoring," Technical Report CSTR 2010-03, Stanford University, Tech. Rep., August 2010.

[4] A. Prayati, C. Antonopoulos, T. Stoyanova, C. Koulamas, and G. Papadopoulos, "A modeling approach on the telosb wsn platform power consumption," *Journal of Systems and Software*, vol. 83, no. 8, pp. 1355 – 1363, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121210000087