

Energy Saving Potential Using Active Networking on Linux Mobile Phones

Kasper Revsbech Janus Heide Kim Højgaard-Hansen Gian Paolo Perrucci Frank H. P. Fitzek
 Electronic Systems Electronic Systems Electronic Systems Electronic Systems Electronic Systems
 Aalborg University Aalborg University Aalborg University Aalborg University Aalborg University
 Email: revsbech@es.aau.dk Email: speje@es.aau.dk Email: kimhh@es.aau.dk Email: gpp@es.aau.dk Email: ff@es.aau.dk

Abstract—This work investigates the deployment of active networking on mobile devices. The objective is to determine the possibility of decreasing energy consumption on mobile devices accessing Internet services, by use of active networking. The approach is to move the TCP/IP stack from the mobile device onto an access point. To enable this a light-weight protocol based on Bluetooth has been implemented to facilitate the link between the mobile device and the access point. A design and implementation that is transparent to user applications and runs on the mobile device is presented. The performed tests indicate a decrease in the energy consumption of 25%.

I. INTRODUCTION

Today mobile devices ship with an increasing number of features in terms of complex software and hardware such as Global Positioning System (GPS), Wireless Local Area Network (WLAN), high resolution camera and/or color display. This extra functionality results in increased energy consumption and a higher heat dispatching. As the battery technology has not undertaken the same rapid development as the mobile devices in general the time between recharges is reduced. Because of the heat generation and the limited energy resources the mobile device manufacturers and research projects are looking into mechanisms to provide the same services but with a reduced energy consumption.

The focus of this work has been energy optimizations in the network stack, specifically the TCP/IP stack used to connect to the Internet, since applications utilizing the Internet are moving to the mobile devices. Two wireless technologies that are often supported by mobile devices are WLAN and Bluetooth. Bluetooth however is still the most used on mobile devices. It is possible to access the Internet through a Bluetooth link to a gateway if the Bluetooth Network Encapsulation Protocol (BNEP) is used. As the name states BNEP provides an encapsulation of network packets enabling TCP/IP traffic on a Bluetooth link [1]. BNEP utilizes the complex TCP/IP stack which is executed on the mobile device. This increases the workload on the application processor and generates a transmission overhead due to the TCP/IP encapsulation which increases the power consumption. As proposed in [2] we follow the idea to move the functionality of the TCP/IP from the mobile device onto the gateway to which the mobile device is connected. The concept is shown in Figure 1. As shown the functionality associated with the TCP/IP protocols is moved to a gateway if possible. Instead application data is

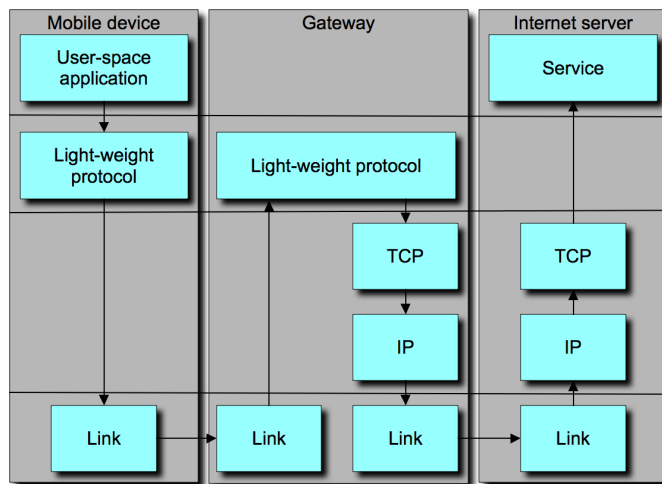


Fig. 1. The concept of a TCP/IP stack moved from the mobile device to a gateway substituted with a light-weight protocol

exchanged between the mobile device and the gateway via a new lightweight protocol. Thus data from an application on the mobile device is forwarded to the gateway where it is converted to TCP/IP traffic, before the packet is transmitted to the external service.

In the presented approach a new layer is added to the network stack on both the mobile device and gateway. Both devices still work as expected with regular network devices but is able to use the improved protocol if this is present at the gateway. To make this possible the mobile device and the gateway need to be able to react according to the type of device connected effectively making the modified network devices cognitive [3]. Cognitive devices are a first step towards the broader term Active Networking where the nodes in the network performs computations on and modify the packet contents [4]. Research in Active Networking is re-surfacing whenever improvements in terms of safety and efficiency within active technology have happened and is currently receiving attention again to enable network devices to adapt more flexible to their environment [3]. In this case the context awareness and flexible adaption of Active Networking means a scenario where the mobile device is able to sense if an access point capable of using a light-weight protocol, as illustrated in Figure 1, is present and if so use this instead of

the traditional TCP/IP protocol.

As shown in Figure 1 a light-weight protocol stack will substitute the TCP/IP stack on the line between the mobile device and the gateway. The Nano IP project [5] has designed a light-weight protocol to use on a Local Area Network (LAN) and shows that the communication overhead of a full TCP/IP stack can be reduced implementing a light-weight protocol.

In this paper we present an approach to decrease the energy consumption when accessing the Internet via Bluetooth. We present an implementation of a light-weight protocol stack based on a GNU/Linux system. The implementation is tested on the Nokia N810 Internet tablet and it is shown that the power consumption can be reduced by approximately 25% depending on the scenario.

The remainder of the paper is organized as follows. In Section II necessary background knowledge is introduced and technical aspects of the proposed solution is outlined. In Section III the test setup and obtained results are presented. The final conclusions are drawn in Section IV.

II. METHOD

As described the motivation behind this work is to investigate the potential of saving energy on a mobile device by relocating the mechanisms of TCP/IP to a gateway that does not suffer from limited battery. This is done by introducing a light weight protocol between the mobile device and the gateway. The reasoning of moving the functionality of the TCP/IP stack from the mobile device to a gateway is that the TCP/IP stack provides complex functionality to facilitate communication on a large scale network e.g. flow control. This functionality is needed in order to utilise services on the Internet but can be performed by a gateway instead of by the mobile device. A light-weight protocol between the mobile device and the gateway can result in energy saving on the mobile device because the computational complexity of executing the protocol stack is reduced and thereby less energy is consumed by the processor. Secondly a simpler protocol can reduce the header size and thereby decrease the amount of data that must be transmitted. Letting the gateway handle the transaction between the light-weight protocol stack and the TCP/IP protocol stack can be seen as a one-hop Active Networking approach.

The use of Active Networking, the design of the light-weight protocol and the functionality added on the mobile device and the gateway to implement the protocol is explained in the following.

A. Active Networking

Active networks are networks where the nodes can perform computations on and/or modify the content of packets. This processing can be customised on a per-user or per-application basis which differs from the traditional networks where a router can change packet headers but not the user data and where routing is independent of the user process or application generating the packets. Two approaches are common in Active Networking (AN), the programmable switch approach and

the capsule approach [4]. The programmable switch approach separates the packet processing from the mechanism to download new programs for processing of the packets. The capsule approach embeds code to be executed at each active node in the packets along with the user data. In this work the programmable switch approach is used and thus new rules of processing packets are downloaded to both the gateway and the mobile device beforehand.

B. Light-weight protocol stack

The goal is to replace the traditional TCP/IP stack on the first hop, by providing only a subset of the functionality in order to achieve a TCP/IP like behaviour. This means that the protocol must provide the following functionality:

- Guaranteed data delivery
- Data integrity
- Data ordering
- Multiple simultaneous flows
- Establish and terminate a TCP/IP connection to an Internet server on demand from an application on the mobile device

As described in the introduction the focus of this paper is on the Bluetooth stack and the TCP/IP stack interface provided by BNEP. Therefore the Bluetooth protocol stack has been investigated to determine if any existing protocol provides the needed functionality. Guaranteed data delivery, data integrity and data ordering can be ensured by using Logical Link Control and Adaptation Protocol (L2CAP) with Asynchronous Connectionless link (ACL) which is designed for packet traffic [6]. L2CAP also provides functionality for multiple flows as it multiplexes multiple logical connections onto the same physical connection. Thus by using the L2CAP protocol on top of an ACL¹ [6] the only functionality that must be provided by the light-weight protocol, is to establish and terminate TCP/IP connections to an Internet server, from the gateway. This can not be done by existing Bluetooth protocols, however a protocol supporting this can be used on top of the L2CAP layer. One such protocol that allow clients to create and terminate TCP/IP connections on a Internet gateway is the Socks V.5 protocol [7]. By combining these existing protocols as shown in Figure 2 a light-weight protocol stack supporting the needed subset of functionality is provided. As seen in the figure using the light-weight protocol the user-space data only has to pass the Socks layer before it reaches the L2CAP layer which is common for both stacks. This means that the header of the data packages is reduced. As the Socks protocol do not add any header to data packages the header size of the light-weight protocol is only 4 B from L2CAP compared to 40 B + 4 B for the full TCP/IP on top of L2CAP.

C. Implementation

An implementation based on the light-weight protocol stack has been created to identify the potential energy saving. The implementation is used to enable measurements on the Nokia

¹L2CAP can only be used on top of an ACL

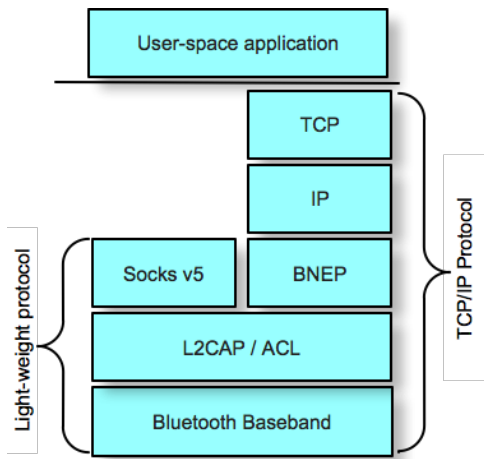


Fig. 2. Components of the light-weight protocol stack where socks V.5 forwards the TCP connection functionality to the gateway and the L2CAP / ACL handles delivery, integrity, ordering and multiple flows to the gateway. Similarly the for the components in the TCP/IP stack where the link to the L2CAP / ACL layer is handled by BNEP

N810 Internet tablet. The N810 is based on a GNU/Linux platform where a lot of open source software is available. This software can be modified and used to facilitate the implementation. The implementation provides the following functionality:

- On the mobile device:
 - A client side implementation of the light-weight protocol
 - A TCP/IP compatible interface towards user-space applications
- On the gateway
 - A server side implementation of the light-weight protocol
 - A bridge between the light-way protocol and the TCP/IP protocol

1) *Mobile device implementation:* The purpose of the mobile device implementation is to present an interface to the light-weight protocol stack that is compatible to that of the TCP/IP interface. In this way unmodified network applications can be executed by using the light-weight protocol stack for communication. The Nokia N810 includes the Bluetooth stack Bluez [8] and therefore Bluez is used to provide the baseband L2CAP functionality.

In Linux the interface from applications to the network stack is provided by the POSIX socket interface and is provided as a part of the Linux kernel functionality by offering the API as system calls. In order to use standard applications on the light-weight protocol stack, the implementation must provide a socket interface to the applications. The tsocks application [9] with some modifications can provide such an interface as the tsocks application works by capturing socket systems calls and change the behaviour to comply with the socks V.5 RFC. The tsocks application works as a wrapping application that translates the relevant system call. As an example when an

application calls connect on a TCP/IP socket tsocks translates this and uses the Socks V.5 protocol to let the gateway call connect. From the applications point of view it is connected to the requested server but in reality it is only connected to the gateway which is connected to the requested server. Thereby the tsocks application fulfills the requirement of providing the same interface to the application which insures that applications can be used without rewriting or relinking. The tsocks application does not provide a L2CAP connection to the gateway and have therefore been modified to do so.

2) *Gateway implementation:* The purpose of the gateway implementation is to provide a socks V.5 compliant server, that can establish and terminate a TCP/IP connection to an Internet service and forward the data between the mobile device and the Internet service. This is the standard functionality of a standard Socks server that implements Socks V.5, therefore such and server is used in this case the SS5 server [10] was chosen. The SS5 server has been altered from using a TCP/IP connection to use the light-weight protocol between the mobile device and the socks server.

III. RESULTS

The goal is to compare the energy usage on the mobile device for the light-weight protocol setup and the TCP/IP protocol stack setup. To achieve this traffic generated by the packet generator Packgen [11] is transmitted using both protocol stacks. The setup consists of the following software and hardware.

- Mobile device:
 - HW Nokia N810
 - SW Operating System RX-44_DIABLO_4.2008.23-14
 - SW Modified tsocks based on tsocks-1.8
 - SW Bluez utils dist_3.23-1
 - SW Packgen-0.2 as client
- Gateway:
 - HW IBM x31
 - SW Operating System Ubuntu Linux 8.04.1
 - HW Belkin USB Bluetooth Adapter model F8T013
 - SW Modified socks server based on ss5-3.6.4
- Internet server:
 - HW IBM R51
 - SW Operating System Ubuntu Linux 8.04
 - SW Packgen-0.2 as server
- Measurement equipment
 - HW Agilent 66319D DC power source
 - HW Modified Nokia N810 battery

Packgen is running on the N810 and generates packets of 640 B as this is near the standard Bluez [8] Maximum Transfer Unit (MTU) of L2CAP². To investigate the impact of multiple simultaneous connections Packgen is configured to generate packets distributed equally on one to seven connections, which is named flows in Packgen. The total data rate is set to 15 KB/s. The flows are transmitted from the mobile device

²L2CAP allows control of the MTU offered to the higher layers

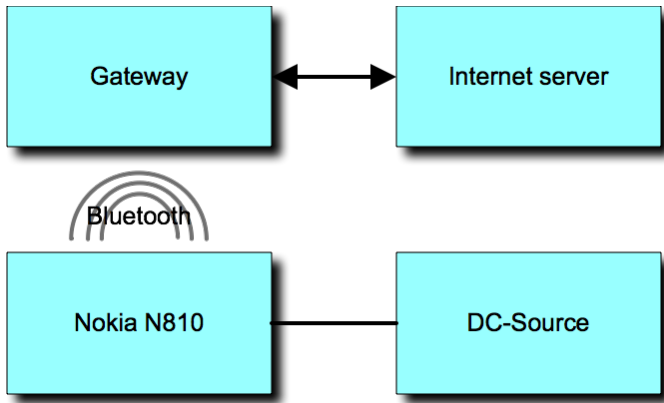


Fig. 3. Test setup where the energy consumed by the N810 can be measure in the DC-Source

to the gateway via Bluetooth. On the gateway the traffic from the mobile device is forwarded through an Ethernet interface to an Internet service.

To determine the energy usage, the voltage and current are sampled from an Agilent 66319D that acts like a power source for the N810. The Agilent samples at 4.096 kHz and calculates the mean for each second by applying a Hamming windowing function [12]. The test setup is illustrated in Figure 3, as shown the Nokia N810 is powered by an Agilent DC-source where the energy usage of the N810 can be measured while it transmits traffic to the Internet service on the Internet server by use of the Bluetooth capable gateway.

Both the light-weight protocol and the TCP/IP protocol are tested with one to seven simultaneous flows resulting in seven test scenarios, one for each number of simultaneous flows. For each test scenario 30 tests have been conducted. One test has a duration of 300 seconds resulting in 4.5 MB transferred data.

For each test run the mean power per second and the total mean power is calculated. To isolate the power used for transmission, the idle power of the N810 is subtracted. The idle power has been measured while the device was not running any applications, and with Bluetooth switched off. The idle power is measured to 328 mW. This power used for transmission is multiplied with the test duration in seconds to determine the total consumed energy during the test run. The total consumed energy is divided with the total transmitted data, to determine the amount of energy consumed per transmitted KB. The consumed energy per KB for both the TCP/IP protocol and the light-weight protocol is plotted in Figure 4 and is calculated with a 95% confidence interval in Table I.

From Figure 4 it can be seen that both protocols both have an increased energy usage when there is more than one flow. This observation holds for a confidence interval of 95%, see Table I. Furthermore it can be seen that the difference between the energy usage of the light-weight protocol stack and the TCP/IP protocol stack is nearly constant.

A. Complexity reduction

In order to quantify the reduction of complexity we have conducted measurements of the CPU utilisation of kernel

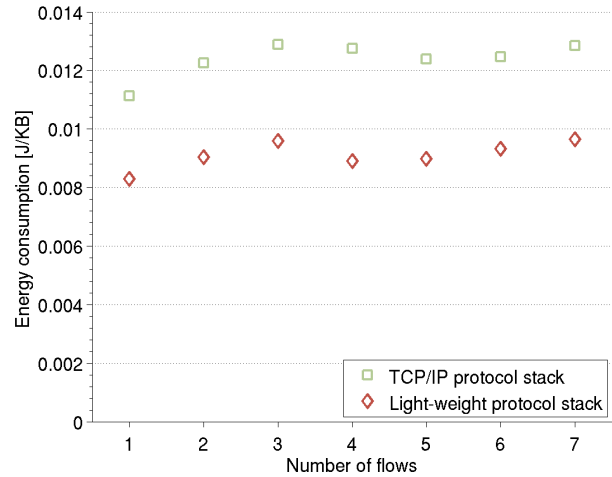


Fig. 4. The mean energy usage for different number of flows, all with a bandwidth of 15 KB/Sec

TABLE I
LOWER AND UPPER BOUND GIVEN BY A 95% CONFIDENCE INTERVAL FOR THE ENERGY USAGE IN JOULE PER TRANSFERRED KILOBYTE FOR DIFFERENT NUMBERS OF SIMULTANEOUS FLOWS FOR THE TCP/IP PROTOCOL STACK AND THE LIGHT-WEIGHT PROTOCOL STACK

TCP/IP protocol stack			
Number of flows	Lower bound	Mean	Upper Bound
1	0.01101883	0.01112194	0.01122506
2	0.01210426	0.01224774	0.01239122
3	0.01264428	0.01288954	0.01313479
4	0.01263602	0.01275694	0.01287786
5	0.01226359	0.01239101	0.01251843
6	0.01231468	0.01247000	0.01262531
7	0.01272226	0.01283830	0.01295433
Light weight protocol stack			
Number of flows	Lower bound	Mean	Upper Bound
1	0.00819546	0.00827901	0.00836256
2	0.00893744	0.00903681	0.00913619
3	0.00946343	0.00958593	0.00970842
4	0.00876245	0.00889159	0.00902073
5	0.00887079	0.00897230	0.00907380
6	0.00911885	0.00931339	0.00950792
7	0.00954007	0.00964948	0.00975888

TABLE II
ENERGY SAVING PER KILOBYTE FOR THE LIGHT-WEIGHT PROTOCOL COMPARED TO TCP/IP FOR DIFFERENT NUMBERS OF SIMULTANEOUS FLOWS

Flows	1	2	3	4	5	6	7
Saving [%]	25.6	26.2	25.6	30.3	27.6	25.3	24.8

processes during transmission. It has been chosen to measure the kernel usage as the protocol stacks are hosted in the kernel in Linux. The scenario is identical with the measurement of the energy consumption for a single flow. In addition to the CPU utilisation of BNEP and L2CAP, the idle CPU utilisation

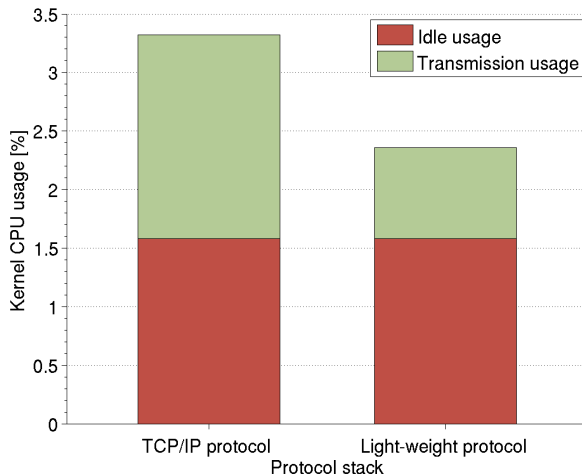


Fig. 5. CPU usage of kernel operations in % for the TCP/IP protocol and the Light-weight protocol

TABLE III
CPU USAGE OF KERNEL OPERATIONS IN % WITHIN A 95% CONFIDENCE INTERVAL

Protocol	Lower bound	Mean	Upper Bound
TCP/IP	3.26	3.31	3.37
Light-weight	2.27	2.36	2.44
None (idle)	1.52	1.58	1.65

is also measured. In this way it is possible to determine any reduction in CPU utilisation due to network transmission only.

The CPU utilisation was measured with sar, which is a part of the sysstat utilities for Linux [13], at a frequency of 1 Hz. BNEP and L2CAP was measured during a transmission of 300 s and the idle cpu utilisation was measured over a period of 300 s, all measure repeated 30 times. The Upper and Lower bounds with a 95% confidence interval along with the mean can be see in Table III.

It is assumed the CPU utilisation in idle mode is constant and due to system tasks that are not related to network transmission. Therefore the idle CPU utilisation is subtracted from the CPU utilisation in BNEP and L2CAP in order to identify the CPU utilisation that is due to executing the network stack, see Figure 5. The left column indicate the total CPU utilisation during a BNEP transfer and the right column indicate CPU utilisation during a L2CAP transmission. The lower portion of the columns indicate the CPU consumed in idle state and the upper section thus indicates the CPU utilisation due to the transmission. If the idle consumption is subtracted it can thus be seen that the CPU utilisation has been reduced to approximately a third using the light-weight protocol instead of the TCP/IP protocol.

IV. CONCLUSION

As seen in Figure 4 and Table I it can be concluded that by using a light-weight protocol instead of TCP/IP between the mobile device and the gateway, energy can be saved.

If for example the scenario with one flow is considered, transmission of 1 KB consumes 25.6% less energy via the light-weight protocol than by the TCP/IP protocol (The saving in percentage can be seen for all the number of flows in Table II).

The overhead for the light-weight protocol is reduced with 40 B, from $640 + 40 + 4 = 684$ B to $640 + 4 = 644$ B which is a reduction of 5.8%. This reduction is significantly lower than the reduction in energy consumption which indicates that the reduced use of baseband is not the only contributor to the energy saving. The reduction in CPU utilisation is significant, see Figure 5, which indicates that reduction in complexity can significantly reduce energy consumption.

The results and energy saving potential presented in this paper are all based on measurements where the mobile device uploads data to the Internet service. Because of the dissimilarity in the task of sending and receiving in the TCP/IP stack we can not conclude that a saving of the same level on download can be obtained.

ACKNOWLEDGMENT

Authors would like to thank Nokia for providing technical and financial support as well as mobile phones to carry out the measurement campaign. Special thanks to Mika Kuulusa, Gerard Bosch, Harri Pennanen, Nina Tammelin, and Per Møller from Nokia. Also a thanks to Ben Krøyer for the support in setting up the measurement equipment. This work was partially financed by the X3MP project granted by Danish Ministry of Science, Technology and Innovation.

REFERENCES

- [1] *Bluetooth Network Encapsulation Protocol (BNEP) Specification*, Bluetooth SIG Inc., 2001. [Online]. Available: <http://grouper.ieee.org/groups/802/15/Bluetooth/BNEP.pdf>
- [2] M. Schläger, B. Rathke, A. Wolisz, and S. Bodenstern, "Advocating a remote socket architecture for internet access using wireless lans," *Mob. New. Appl.*, vol. 6, no. 1, pp. 23–42, 2001.
- [3] T. Arildsen and F. H. Fitzek, "The c-cube concept - combining cross-layer protocol design, cognitive-, and cooperative network concepts," in *Cognitive Wireless Networks*, F. H. Fitzek and M. D. Katz, Eds. P.O. Box 17,3300 AA Dordrecht, The Netherlands: Springer, 2007.
- [4] T. D. et al., "A survey of active network research," *Communications Magazine, IEEE*, vol. 35, no. 1, pp. 80–86, jan. 1997.
- [5] Z. Shelby, P. Mahonen, J. Riihijarvi, O. Raivio, and P. Huuskonen, "Nanoip: the zen of embedded networking," *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 2, pp. 1218–1222 vol.2, May 2003.
- [6] *BLUETOOTH SPECIFICATION Version 2.1 + EDR*, Bluetooth SIG Inc., 2007.
- [7] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones, "SOCKS Protocol Version 5," RFC 1928 (Proposed Standard), Mar. 1996. [Online]. Available: <http://www.ietf.org/rfc/rfc1928.txt>
- [8] "Bluez, official linux bluetooth protocol stack." [Online]. Available: <http://www.bluez.org>
- [9] "Tsocks." [Online]. Available: <http://tsocks.sourceforge.net/>
- [10] M. Ricchetti, "Ss5." [Online]. Available: <http://ss5.sourceforge.net/>
- [11] "Packgen." [Online]. Available: <http://packgen.rubyforge.org/>
- [12] *USER'S GUIDE, Model 66319B/D, 66321B/D, Mobile Communications DC Source*. [Online]. Available: <http://www.home.agilent.com>
- [13] *Sysstat*. [Online]. Available: <http://pagesperso-orange.fr/sebastien.godard/>