Power Consumption Overhead for Proxy Services on Mobile Device Platforms

Troy Johnson
Department of Computer Science
Central Michigan University
Mount Pleasant, MI 48859
johns4ta@cmich.edu

Patrick Seeling
Department of Computer Science
Central Michigan University
Mount Pleasant, MI 48859
pseeling@ieee.org

***Abstract*—As mobile data consumption increases, several venues of research investigate energy efficiency and optimizations, oftentimes realized through on-device services; optimization approaches based around on-device optimizations in turn need to overcome the power consumption added by these services. We present a measurement framework for the power consumption of mobile devices based on the Pandaboard (mobile) development platform using the Android operating system. Utilizing the framework, we perform power consumption evaluations for basic web requests as well as HTML5 video streaming to a mobile browser in (i) direct connection and (ii) indirect connections through a mobile proxy server on the device, which represents a baseline for application layer optimization approaches.**

We find that using the local proxy server on the device for web requests results in very limited overheads, both in terms of power consumption and added delays which tend to remain request size independent. HTML5 video streaming through the proxy service incurs an average energy consumption penalty of approximately one percent. Our findings corroborate that even smaller realized optimization gains through services on mobile devices themselves are not usurped by the service enablers consuming mobile CPU cycles.

***Index Terms*—Mobile communication; Middleware; Energy consumption**

I. Introduction and Related Works

In recent years, the amount of data that mobile users consume has increased significantly. Current predictions by Cisco, Inc. indicate a continuation of this trend for years to come [1]. As the network interfaces of mobile devices typically consume most of the limited battery power available, see, e.g., [2], the correlation between the amounts of data, related user interactions and limited battery power has spurred research efforts that investigate the possibilities of energy efficient mobile data delivery. As indicated by prior research efforts, mobile applications feature characteristic behaviors with respect to their network utilization. The periodic nature of mobile application updates has attracted specific attention, see, e.g., [3].

Several different venues of research target varied levels of optimization and required network provider infrastructure to realize potential energy savings while simultaneously following quality of service restrictions; the goal is to balance user experience and a mobile device's charge longevity. The CasCap approach outlined in [4], for example, utilizes mobile device clones in the cloud to optimize the network traffic. An immediate benefit here is the outsourcing of most computational overhead into the cloud. A recent overview of challenges that mobile cloud computing faces and potential solutions can be found in, e.g. [5], [6]. Similarly, in [7], the authors provide a detailed survey of current approaches to energy efficient delivery of multimedia to mobile devices. A plethora of approaches to mobile connectivity and computation optimizations (focusing on battery life and spectral efficiency issues) have in common that they commonly rely on an additional middleware component(s) executed on the mobile device.

This middleware typically acts as an on-device proxy service to realize benefits or enable new interaction paradigms, such as display networks [8] or mobile content sharing [9], [10]. When regarding approaches that are based on proxy services, we note that they originally date back to the advent of "Green IT," with a focus on wired networking, see, e.g., [11] and [12] for a recent overview. In addition, a proxy-based approach allows for potentially saving energy when utilizing heterogeneous networks while mobile by intelligently switching between them without interrupting an ongoing stream, see, e.g., [13]. In addition, a standards-based and application-transparent proxying based on the SOCKSv5 standard was presented in [14]. Furthermore, content control applications that utilize a mobile proxy server as mitigation method for web browsers and web-based applications (similar to the well-known "Net Nanny") were recently introduced as well, see, e.g., [15].

As with any optimization approach that is executed on a mobile device itself, however, an overhead is incurred due to increased processing and storage burdens. In turn, optimizations that require a large number of mechanisms implemented on the mobile device itself need to initially overcome these potential penalties before they can yield (e.g., power) savings benefits. While commonly available measurement values for mobile device components exist, see, e.g., [16], performance measurements that relate these to real-world performance are underrepresented in the literature.

In this paper, we evaluate the middleware-introduced power

978-1-4799-2355-7/14/$31.00 ©2014IEEE

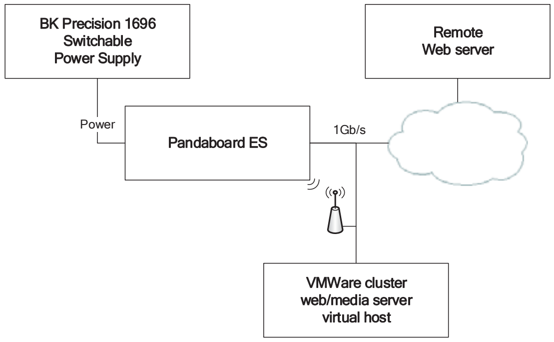

Fig. 1. Overview of the measurement setup with switchable power supply, development board, and remote servers.

consumption impact on the power consumption in mobile devices that optimization schemes need to overcome. We base our evaluation on a mobile SOCKSv5 standard proxy service which we adapted to be used with Android devices. The remainder of this paper is structured as follows. In the following section, we review our measurement setup and performance evaluation metrics. In Section III, we present results for frequent fixed destination web requests for a wired LAN setup as an initial baseline as well as for wireless LAN configurations. We evaluate the impact of the web request sizes for wireless LANs in the succeeding section. We present results for video streaming in Section V. We conclude in Section VI with an outlook on future activities.

II. Methodology and Metrics

In this section, we outline our setup and evaluation methodology. We differentiate different network scenarios for which we evaluate the mobile proxy server, namely (i) utilizing an Ethernet network connection and (ii) a wireless LAN network connection.

A. Measurement Setup

At the core of our setup, we utilize a Pandaboard ES mobile software development platform, which features a Texas Instruments OMAP 4460 dual core ARM Cortex-A9 processor with 1 GB of DDR2 RAM, SMSC 10/100 Mbps Ethernet port, and LS Research WLAN/Bluetooth wireless module, next to other components. (Please refer to http://www.pandaboard.org for more details.) We employ the open-source Android distribution (version 4.1.2, 'Jelly Bean') as the operating system software. We illustrate our overall measurement setup in Figure 1. The Pandaboard is powered by a BK Precision 1696 switchable power supply, which features serial port access to read out voltage and ampere values over time. We connect the power supply to a Linux desktop computer serial port and timestamp the values obtained over time to measure the power consumption incurred by the Pandaboard. The Pandaboard is connected through a 1 Gbps maximum speed Ethernet campus network, which eliminates potential bottlenecks. For wireless measurements, we utilize an externally connected WLAN antenna, which connects to the campus network through a dedicated WLAN access point, again eliminating bottlenecks for the amounts of data we consider throughout. We additionally note that a combination of input devices and an external monitor were connected as well.

On the server side, we utilize a locally hosted virtual machine next to Internet-routed web requests. The local server employs the Debian Linux distribution as operating system with the apache2 HTTP server and the popular Video Lan Client (VLC) as media streaming application. We stream a pre-encoded video sequence, the popular open-source movie *Tears of Steel* (see http://www.tearsofsteel.org for more information), utilizing HTML5 video streaming. The video-only sequence was transcoded offline into a resolution of 864×480 at 24 frames per second in the Theora video codec and encapsulated using the OGG container format, both commonly utilized for HTTP video streaming "on the web" and suitable for mobile playout. The resulting video bit stream has a duration of 12 minutes, 14 seconds and an average bit rate of 1.42 Mbps. The bit rate in turn falls well within range of the network bandwidth capacity.

B. Mobile SOCKSv5 Proxy Server

Several implementations of the SOCKSv5 standard [17] exist to date, which allow utilization of a remotely hosted standard-conforming SOCKSv5 proxy server (typically from a desktop computer through an organizational server). Mobile implementations, however, are less frequent. One example of an implementation for the Android operating system is the anonymity generating Orbot application (see http://www.torproject.us for more details), which routes traffic into the TOR network and contains "proxification" methods for applications as well (i.e., transparently forcing the usage of the proxy through, e.g., modifications of the iptables firewall). We generated a basic Android service application that is based on the jSOCKS proxy server implementation [18], which is open-source (entirely written in JAVA) and does not require any privileges, such as root level system access. As the service is executed within the Dalvik VM utilized on Android devices, it incurs a minimal computational overhead when compared to native applications. This approach, however, is commonplace to allow broadest application compatibility and encouraged for developers of the platform.

C. Performance Metrics

In the following, we briefly outline the metrics used to evaluate the performance of either scheme. Initially, we capture the reported voltage level $v(t_l)$ [V] and the current $i(t_l)$ [A] as reported by the power supply and timestamped at time t_l on the connected desktop computer. We similarly calculate the instant power consumption as $p(t_l) = v(t_l) \cdot i(t_l)$ [W]. As the reported values are instantaneous snapshots in time from $l = 0$ at $t(l) = 0$ (denoting the first measurement) to $l = L$, which happened at $t(l) = T$ (whereby T denotes the last measurement), we calculate the time passed between consecutive measurement instances as $t(l) = t_l - t_{l-1}$. To determine the energy that was used in the l-th measurement

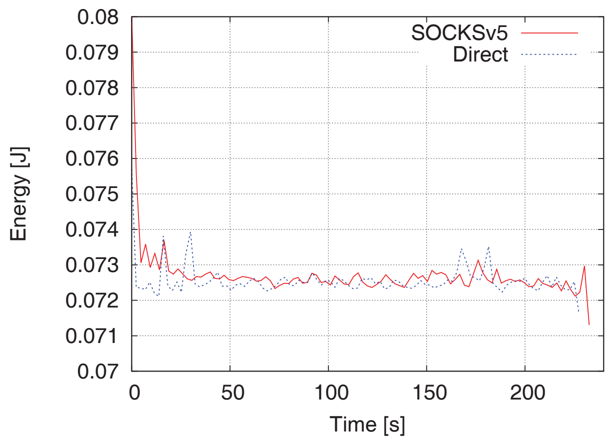

Fig. 2. Energy consumption while performing 100 web requests directly or through a mobile SOCKSv5 service, smoothed over time.

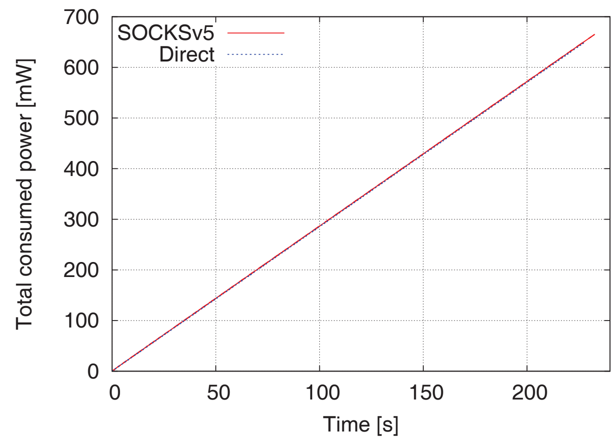

Fig. 3. Compounded energy consumption for performing 100 web requests directly or through a mobile SOCKSv5 service.

period, we calculate $e(l) = t(l) \cdot w(t_l)$ [J]. We denote the energy that was used in a measurement period up to $t(l)$ as $E(l) = \sum_{m=0}^{l} e(m)$ and the average as $\overline{E} = \frac{1}{L}E(L)$.

We calculate the relative overhead that stems from the mobile proxy server usage as

$$o = \frac{\overline{E}_{\text{Proxy}} - \overline{E}_{\text{Direct}}}{\overline{E}_{\text{Direct}}}. \quad (1)$$

III. Performance Evaluation for Web Requests

To perform a representative evaluation of frequent HTTP web requests, we utilize requests for Google's home page. The goal of this particular measurement scenario is to evaluate the performance impact of frequent requests through the local proxy server, which has to perform the additional connection tasks each time a request is made. We developed a direct measurement application for Android, which will request `http://www.google.com` without further resolving any HTTP objects within. Individual requests are followed by a sleep period of 2 seconds for both, direct requests and requests through the mobile SOCKSv5 proxy service. As requests are made without utilizing a browser, no caching is involved client-side.

A. Fixed Network

In the fixed network scenario, the Pandaboard is connected through wired Ethernet to the campus network while performing the web requests. The requests typically coincide with high power consumption levels, as illustrated in Figure 2 for an exemplary 100 web requests with both configurations. We observe that both approaches exhibit an initial "spike" behavior and an otherwise low level (with some general noise due to overall device activities). There is no immediately visible trend for the momentary power consumptions, as in both approaches, there are several bursty periods of slightly elevated consumptions on top of the actual web requests.

Next, we evaluate the total (compounded) energy consumption that is observed when performing these requests for a certain period of time. We illustrate 100 subsequent requests directly and through the mobile proxy server in Figure 3. We initially observe that despite the short-term fluctuations, overall we note a steady, linear increase in the energy consumed while using either approach. More significantly, we do not witness an immediately significant difference between the approaches, which is indicated by the almost indistinguishable values in the plot. Lastly, we compare the overhead between the approaches numerically in Table I, where we analyzed the 300 highest levels of energy consumption measured for periods of placing 300 requests. We note that the direct approach results in a higher average level of energy consumption (albeit with a larger variability), whereas the proxy-based approach yields a lower average and variability of energy usage values determined. Overall, this results in an overhead of $o = -0.0563$, which presents an initially counter-intuitive result. (Differently worded, by utilizing the mobile proxy server consuming additional CPU cycles, potential energy savings of 5.6% could be realized without requiring any additional modifications.) Taking into account that partially significant variability exists due to some outliers in the total duration (as some measurement points can exhibit significant delays or coincide with other unrelated system activities), we note that both approaches are very similar.

TABLE I
Summary values for 300 direct and proxy-routed web requests over traditional Ethernet and wireless LAN networks.

Interface	Approach	Average [J]	Standard Deviation [J]	Confidence Interval (99%)
LAN	Direct	0.0912	0.0139	0.0021
	Proxy	0.0860	0.0062	0.0009
WLAN	Direct	0.0900	0.0125	0.0019
	Proxy	0.0902	0.0089	0.0013

B. Wireless Network

Shifting our evaluation to HTTP requests made over the wireless network interface, we present our results in Table I. (We note that a graphical evaluation would yield results similar to those presented in Figures 2 and 3.) We initially note that

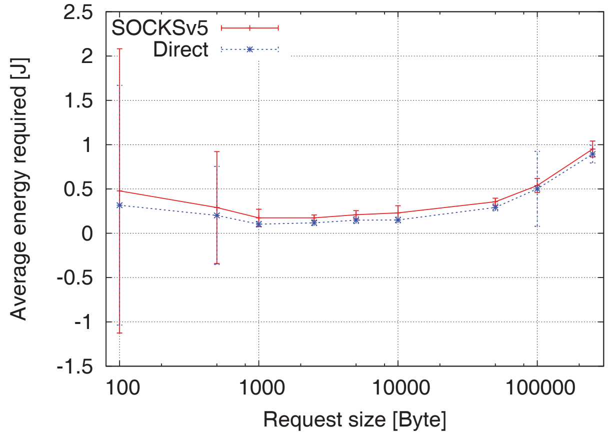

Fig. 4. Average energy consumption and standard deviation for requesting different amounts of data from a local server using a direct or mobile SOCKSv5 proxy server.

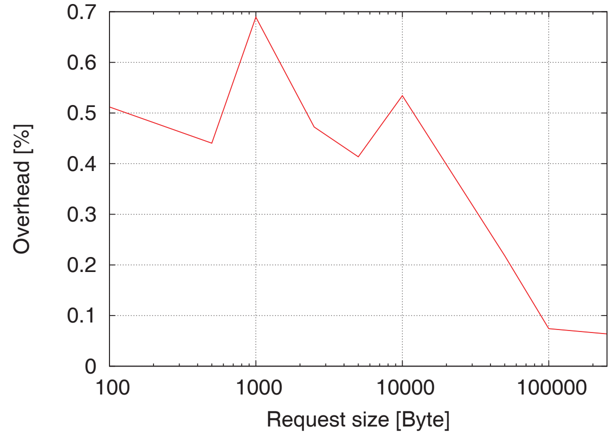

Fig. 5. Average energy consumption overhead for requesting different amounts of data from a local server as in Equation 1.

both approaches are very close with respect to their measured average energy consumption, resulting standard deviations, and narrow confidence intervals.

Comparing these results with those obtained for the Ethernet scenario, which outlines the base case without active wireless communications overheads, we do not note a significant difference in the average energy usage for the individual web requests.

IV. Impact of Web Request Variations

Motivated by the closeness of requests, we now more closely evaluate the impact of the web request size over a wireless LAN on the overall power consumption. To limit the impact that external networks can have (such as different delays), we perform this direct performance comparison within the on-campus VM environment illustrated in Figure 1. A dedicated virtual machine uses the `apache2` web server and hosts a Python script that generates a requested number of bytes, additionally eliminating potential caching impacts. We perform 100 repeated measurements for each different web request size and delete significant outliers.

A. Power Consumption

Initially, we illustrate the average energy used per different request size in Figure 4. We immediately observe that the mobile SOCKSv5 proxy approach always incurs a penalty over the direct connection, which is readily explained by the additional local processing overhead on the device. Furthermore, we note that both depicted connection methods follow a "slump"-like behavior. Taking the overall variability of measurements into account, until the amount of requested data approaches the single packet payload region, the lowest energy usage is recorded. This initial behavior can be explained by the overhead to establish the initial server connection, which is dominant for the small request size regions. With a further increase of the payload to very large sizes, the number of required transmissions drastically increases, which now accounts for the majority of the energy consumption (rather than the initial connection setup). This increase in the number of packets is now causing longer periods of energy-expensive network activities, causing the rise in energy consumption illustrated. The overall absolute difference between the employment of a mobile proxy and direct connection, however, remains fairly constant, which indicates that the processing overhead for local packet processing prior delivery to an application is rather negligible.

Next, we illustrate the overhead incurred by the mobile proxy server (as in Equation 1) in Figure 5. We initially observe an overall "hump"-like behavior which decreases as the request sizes increase. Part of this behavior is explained by the rather large variability (which we do not illustrate here for clarity) in the regions of smaller request sizes. We explain the overall diminishing overhead with an increase in the web request sizes with the actual overhead in the processing and local loop-back data connections for the proxy service. As the proxy service requires an initial connection setup before performing the pass-through connection from the remote server, additional time and processing resources are required before the connection can be fully passed through the proxy. Both resources incur processing and delay penalties which in turn have negative power consumption impacts. Larger request sizes result in increased numbers of packets to transmit, which "smoothes" the fixed setup overhead over more packets, which results in the decreasing overhead. At 250 kB of requested data, we note a reduction to just below six percent.

Overall, we conclude that based on our measurements, the implementation of a mobile proxy server directly on a mobile device does not result in considerable additional power demands for larger-sized web requests, which are common today.

B. Delay Difference

With most interactive settings, the overhead in terms of additional delays can have a negative impact on user utility

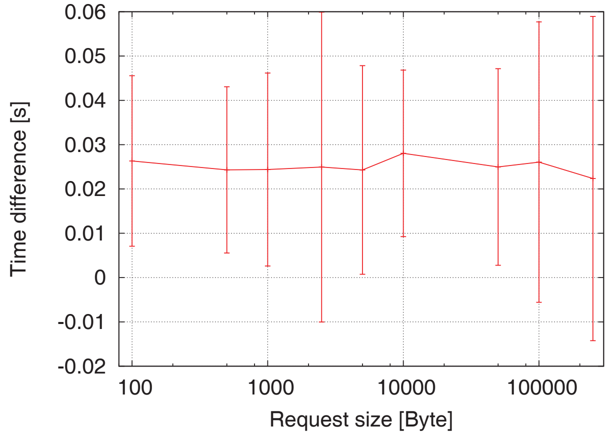

Fig. 6. Average time difference and standard deviation between the direct and the proxy approach for requesting different amounts of data from a local server.

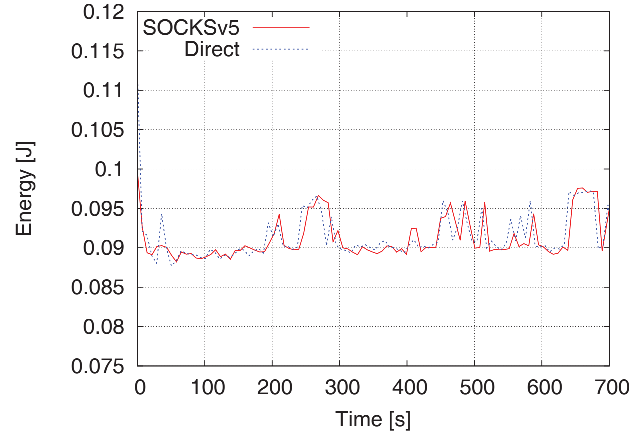

Fig. 7. Energy consumption while performing the video streaming of the *Tears of Steel* open-source video sequence directly and through a mobile SOCKSv5 proxy server, smoothed over time.

or perceived quality of service. In networked mobile settings, increased delays additionally have a negative battery impact through the direct correlation between the two. We illustrate the average delay differences between the proxy and the direct connections for different request sizes in Figure 6. We initially observe a fairly steady level of added average delays to the mobile proxy service in the region of 25 ms. We note some higher levels of standard deviation towards the larger amounts of data requested; an overall maximum of about 37 ms can be noted for requesting 250 kB of data. The variability for the delays is a combination of the script producing the larger chunks of data with now slightly increased variability, as well as the increased number of packets sent for each approach incurring an additional network delay variation.

Overall, we conclude that the observed low level of additional delay represents the fairly constant overhead of setting up the proxy service on the mobile device and processing the initial connection requests.

V. Media Streaming

In this section, we evaluate the impact of a long duration connection through the mobile SOCKSv5 server, in contrast to the previous evaluation of frequent connection requests. Here, a continuous data stream needs to be forwarded to a local application, with the initial setup overhead becoming negligible. The playback of the web video on the Pandaboard is performed using the Firefox for Android web browser. For measurements of the incurred proxy overhead, the browser is reconfigured to utilize the local SOCKSv5 proxy service, similar to the previous web request scenarios. The HTML5 video is in turn displayed on the browser for the entire duration.

A. Fixed Network

We illustrate the sampled power consumption for video streaming over a wired Ethernet network in Figure 7. We observe that most of the measured power consumption values

TABLE II
Summary values for HTML5 video streaming sessions of the open-source *Tears of Steel* movie (video-only) over an Ethernet and a wireless network.

Interface	Approach	Average [J]	Standard Deviation [J]	Confidence Interval (99%)
LAN	Direct	0.0917	0.0051	$7.81 \cdot 10^{-5}$
		0.0913	0.0050	$7.77 \cdot 10^{-5}$
	Proxy	0.0912	0.0052	$7.99 \cdot 10^{-5}$
		0.0912	0.0052	$7.95 \cdot 10^{-5}$
WLAN	Direct	0.0844	0.0047	$7.87 \cdot 10^{-5}$
		0.0847	0.0049	$8.26 \cdot 10^{-5}$
		0.0844	0.0050	$8.35 \cdot 10^{-5}$
		0.0846	0.0056	$9.37 \cdot 10^{-5}$
	Proxy	0.0849	0.0052	$8.65 \cdot 10^{-5}$
		0.0854	0.0051	$8.53 \cdot 10^{-5}$
		0.0849	0.0047	$7.83 \cdot 10^{-5}$
		0.0855	0.0051	$8.50 \cdot 10^{-5}$

fall into a range between 3.25 W and 3.75 W for both approaches. Both approaches additionally exhibit overall periods of heightened power consumption, e.g., from about 250s to about 290s. This time frame coincides with a fast camera movement following an actor (very high level of background content change, motion) and represents a more challenging video decoding task, in turn leading to higher power consumption. Comparatively, however, we observe no significant differences between the two approaches. The aggregated energy consumption resulting from media streaming results in a linear behavior, which additionally is almost indistinguishable between the two approaches as observed for the prior web requests scenario. As indicated by the comparable energy consumption over time in Figure 7, the additional video decoding power requirements let the overall power consumption differences remain minor. We in turn omit a visual representation that closely resembles Figure 3 due to space constraints. We present the aggregated performance values for this scenario in Table II. We note that the average energy used is very similar for either approach, with slightly elevated values for the direct approach. Taking the standard deviation and the very narrow 99% confidence intervals into account, we note that the proxy-based streaming incurs no significant penalties. Comparing this streaming example to the individual web requests, we note that overall, the proxy-based approach now is at the same overall level that we observed for the direct case.

B. Wireless Network

As in the previous web request scenario, we additionally investigate the alternative approach whereby the video streaming is now utilizing a local wireless network. We present the measurement results in Table II for equal numbers of measurement points. (We note that for means of better comparison, we limit the number of measurement points to be equal.) We note that the overall averages are fairly close and furthermore characterized by relatively narrow confidence intervals. The resulting overhead, when comparing the averages of the direct and proxy-based video streaming is 0.0081, or less than one percent. Comparing the overall WLAN levels to the wired measurements, we note a decrease in the overall power consumption. We explain this behavior with the full availability of all network interfaces, whereby the employed operating system seems unable to properly put all different interfaces into a no-power state.

Overall, we conclude that the introduction of the mobile proxy server has only limited negative impacts on the power consumption of about one percent when streaming multimedia due to the significantly higher impact of the decoding processes on the energy consumption and the negligible packet processing overheads.

VI. Conclusion

We approximated the energy consumption overhead of mobile optimization frameworks through a mobile SOCKSv5 proxy server as a low-end baseline. The selected implementation is based on JAVA and performs all network traffic forwarding on the application layer within the Dalvik VM for Android devices. We find that the overall usage of a proxy service on the mobile device in a web request scenario does not incur any significant power consumption penalties. For HTML5 video streaming (continuous packet processing), we note an overhead of about one percent. We determined that any existing overhead is relatively constant and explained through the initial communication setup overheads, rather than on-device processing overheads, which we derive at around one percent. A basic optimization framework would in turn only need to overcome this very low overhead to enable energy savings potentials.

In future research avenues, we plan to investigate further energy savings potentials based on mobile device proxying, in combination with caching and application layer content awareness in tandem with a remote server. Additionally, we are interested in transparent application optimization from an upstream perspective.

Acknowledgment

The authors thank A. Sampson and R. Kohvakka for helping with the measurement environment. This work was supported in part by an Early Career grant from the Office of Research and Sponsored Programs at Central Michigan University.

References

[1] Cisco, “Cisco visual networking index: Global mobile data traffic forecast update, 2011–2016,” Feb. 2012.

[2] A. Carroll and G. Heiser, “An analysis of power consumption in a smartphone,” in *Proceedings of the 2010 USENIX Annual Technical Conference (ATC)*. Boston, MA: USENIX Association, 2010, pp. 21–34.

[3] F. Qian, Z. Wang, Y. Gao, J. Huang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, “Periodic transfers in mobile applications: network-wide origin, impact, and optimization,” in *Proceedings of the 21st international conference on World Wide Web*, ser. WWW ’12. New York, NY, USA: ACM, 2012, pp. 51–60.

[4] Y. Xiao, P. Hui, P. Savolainen, and A. Ylä-Jääski, “Cascap: cloud-assisted context-aware power management for mobile devices,” in *Proceedings of the second international workshop on Mobile cloud computing and services*. ACM, 2011, pp. 13–18.

[5] L. Lei, Z. Zhong, K. Zheng, J. Chen, and H. Meng, “Challenges on wireless heterogeneous networks for mobile cloud computing,” *Wireless Communications, IEEE*, vol. 20, no. 3, pp. –, 2013.

[6] N. Fernando, S. W. Loke, and W. Rahayu, “Mobile cloud computing: A survey,” *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84 – 106, 2013.

[7] M. A. Hoque, M. Siekkinen, and J. K. Nurminen, “Energy Efficient Multimedia Streaming to Mobile Devices – A Survey,” *IEEE Communications Surveys & Tutorials*, vol. 99, no. 99, pp. 1–19, 2012.

[8] N. Davies, M. Langheinrich, R. Jose, and A. Schmidt, “Open display networks: A communications medium for the 21st century,” *Computer*, vol. 45, no. 5, pp. 58–64, 2012.

[9] P. Seeling, “Caching proxying for mobile users,” in *Proc. of IEEE Vehicular Technology Conference (VTC)*, Dresden, Germany, Jun 2013.

[10] ——, “An overview of energy savings potentials through mobile forwarding proxy framework,” *International Journal of Ad Hoc and Ubiquitous Computing*, in print, 2013. [Online]. Available: http://patrick.seeling.org/publication-pdfs/jrn/SE13_IJAHUC_Proxy.pdf

[11] M. Jimeno, K. Christensen, and B. Nordman, “A network connection proxy to enable hosts to sleep and save energy,” in *Proceedings of the IEEE International Performance, Computing and Communications Conference (IPCCC)*, Phoenix, AZ, USA, dec. 2008, pp. 101–110.

[12] A. P. Bianzino, C. Chaudet, D. Rossi, and J.-L. Rougier, “A survey of green networking research,” *IEEE Communications Surveys & Tutorials*, vol. 14, no. 1, pp. 3–20, 2012.

[13] D. Maltz and P. Bhagwat, “Msocks: an architecture for transport layer mobility,” in *Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 3, mar 1998, pp. 1037–1045.

[14] S. Haase and P. Seeling, “Sockx - an application layer network switching framework using socksv5 protocol extensions,” in *IEEE International Conference on Electro/Information Technology (EIT)*, Mankato, MN, USA, May 2011.

[15] R. Bott, “Mobile device usage control in a mobile network by a distributed proxy system,” Patent (provisional) 13/560 092, 2013.

[16] G. P. Perrucci, F. H. P. Fitzek, and J. Widmer, “Survey on energy consumption entities on the smartphone platform,” in *Proc. IEEE 73rd Vehicular Technology Conf. (VTC Spring)*, 2011, pp. 1–6.

[17] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones, “Socks protocol version 5,” RFC 1928 (Proposed Standard), Internet Engineering Task Force, Mar. 1996.

[18] K. Kouzoubov, “Java socks proxy,” 2011. [Online]. Available: http://sourceforge.net/projects/jsocks/