Smart Proxying: an Optimal Strategy for Improving Battery Life of Mobile Devices


Published in:
Proceedings of 2013 International Green Computing Conference (IGCC)

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
Link to publication record in Queen's University Belfast Research Portal

Publisher rights
© 2013 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

General rights
Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.
Smart Proxying: An Optimal Strategy for Improving Battery Life of Mobile Devices

Raffaele Bolla*, Maurizio Giribaldi†, Rafiullah Khan* and Matteo Repetto+

*DITEN Dept. University of Genoa, Via Opera Pia 13, 16145 Genoa, Italy, Email: {raffaele.bolla, rafiullah.khan}@unige.it
†Infocom s.r.l, P.zza Alessi 2/7, 16128 Genoa, Italy, Email: maurizio.giribaldi@infocomgenova.it
‡CNIT, Research Unit of University of Genoa, Via Opera Pia 13, 16145 Genoa, Italy, Email: matteo.repetto@cnit.it

Abstract—Improving energy efficiency of mobile devices is one of the main challenges to extend the battery life. The Wireless Network Interface Card (WNIC) or 3G is usually considered as primary source of energy consumption. To reduce the energy consumption, different proxying schemes have been proposed in literature mostly focusing on increasing WNIC idle periods through traffic shaping, reducing number of bits transmitted/received through compression techniques and putting WNIC or even the mobile device itself into sleep mode whenever possible. This paper presents the possible smart proxying schemes for mobile devices and particularly focuses on the Network Connectivity Proxy (NCP) which impersonates the mobile devices during their sleep periods and maintains their network presence. Till now the proxying schemes have only been considered for static networks without taking into account the mobility of mobile devices. This paper proposes possible proxy architectural designs in order to support host’s mobility. Further, challenges and issues with each architectural design and possible solutions to tackle with those challenges have also been addressed.

Keywords—Smart proxy; network connectivity; energy efficiency; battery life; mobile devices.

I. INTRODUCTION

Today, the energy efficiency especially for battery powered mobile devices has become one of the hot research topic due to their high penetration rate in the world. The users interest in mobile devices e.g., laptops, tablet PCs, smart phones etc is continuously increasing due to the portability and mobility ease that they can offer. Furthermore, the mobile devices are increasing becoming more and more affordable and powerful and run many different type of applications and services. Mobile devices are usually equipped with power-efficient components (CPU, hard disk, display), but this is not enough to extend battery lifetime over a few hours [1]. Further, many applications demand continuous Internet access and transmission over WiFi or 3G contributes to a large share of overall battery consumption [2].

Recently, the proxy based solutions have been proposed to improve the battery life of mobile devices [1], [2]. Unfortunately, all of them only focused on mobile devices attached to fixed local network infrastructure without taking into account their mobility issues. This paper extends our ongoing work on Network Connectivity Proxy (NCP) for fixed network devices by considering also its applicability for mobile device [3], [4], [5]. It has addressed different issues and challenges related to host’s mobility and proposed possible architectural designs to tackle with those issues.

The rest of the paper is organized as follows. Section II briefly presents the related work. Section III presents some possible smart proxying schemes for improving battery life of mobile devices. Section IV presents brief overview of NCP and its possible architectural designs. Section V discusses the applicability of NCP for mobile devices, discusses issues and challenges and proposes possible architectural designs to tackle with them. Finally, section VI concludes the paper.

II. RELATED WORK

There are multiple complementary approaches to improve the battery life of mobile devices. Hardware based approaches focus on the design of energy efficient silicon and optimization of circuitry for improving the energy efficiency. Most of the literature work focuses on the software based approaches which includes (i) putting WNIC in low power mode during idle periods [6], (ii) powering down the primary WNIC but using an additional low power interface that will stay active to quickly revive communication [7], (iii) using the transmit power control [8].

A stateful proxy based ‘Scepter’ design has been proposed in [9]. It tries to maintain same quality of service while reducing number of control and data bits transmitted over WiFi link. The dynamic power aware scheduling for mobile devices using a transparent proxy has been proposed in [2]. The proxy maintains a separate connection to both the client and the server and uses buffers to schedule transmission in bursts to all active clients. A similar approach is proposed in [6] which has considered proxy for delay sensitive applications e.g., Internet radio and YouTube and delay tolerant applications e.g., web browser. Further, traffic shaping proxy has also been proposed in [1] for web applications (also embeds compression techniques and data transfer strategies). A system called ‘O-Sleep’ is proposed in [10]. It smartly puts the user interface instead of processor into sleep mode when the device is processing user request and wakes it up when results are available.

A more challenging proxy based solution has been proposed in literature that allows network host itself to transition into sleep mode and proxy running on home gateway
or on other network host maintains presence on its behalf [3], [11], [12]. Till now all proxying schemes neglect the movement of mobile devices and consider them to be always located in local network. A lot to issues arises if mobile device goes out of local network. This paper extends our previous work on smart proxying [3], [4], [5] by addressing the issues related to the mobility of mobile devices and proposes possible solutions to tackle with such issues.

III. SMART PROXYING FOR MOBILE DEVICES

Smart proxying can be adopted in the following ways:

A. Traffic Scheduling Proxy

The Wireless Network Interface Card (WNIC) or 3G connection often causes the largest portion of power drain in mobile devices. Applications such as multimedia streaming and P2P file download receive packets quite frequently thus requiring WNIC to always stay in active mode. Thus, a Traffic Scheduling Proxy (TSP) can be designed to reduce the energy consumption by shaping the network traffic and transmitting it in the bursts at regular intervals [2], [6]. The generic scenario is shown in Fig. 1 where TSP can be considered running on the home gateway. The network host as well as the TSP buffer certain number of packets before transmitting in bursts. Thus, the host’s WNIC transmitter/receiver can be put into low power state during burst intervals which consumes much less power. To achieve better savings, the TSP will synchronize burst transmission to all of its client devices. The TSP needs to inform each of its clients about the length and order of burst.

B. Data Compression Proxy

Always ON and always available reliable Internet has become part of our daily life. The batteries of mobile devices run-out very quickly when downloading large files over high speed Internet. Thus, a Data Compression Proxy (DCP) can be designed that embeds lossless compression techniques along with data transfer strategies [1]. The content compression is applied between the mobile device and the DCP while data between DCP and remote host over the Internet is uncompressed as shown in Fig. 2. The deployment of DCP can have a dominant effect on mobile device and its WNIC will consume less power as it has to transmit/receive fewer bytes due to compression. To increase the idle periods of WNIC, the DCP can be configured to download and buffer portion of files and then transfer to mobile device over a high speed link. Further, an efficient integration of DCP along with TSP will considerably improve the battery life of mobile devices.

C. Battery Aware Proxy

Usually, the mobile devices are running many Internet-based applications that stays always active in the background and continuously transmit/receive data over the Internet. This implies workload on the processor and always active WNIC causing significant energy consumption. Under critical conditions, the users usually don’t want some background running applications such as weather forecast, google goggles, dropbox etc in order to increase the battery life. Thus, a Battery Aware Proxy (BAP) can be designed that will run on the mobile device and will let the user to profile applications with the battery information. The BAP will switch OFF/ON the application when the battery level crosses the specified threshold for that specific application. Further, the TSP and DCP functionalities can also be linked with a battery threshold [2].

D. Network Connectivity Proxy

Usually, network hosts are idle most of the time but still kept powered-up 24/7 just to maintain the network connectivity for remote access and to maintain the status of applications (e.g., Instant Messenger (IM), Voice-over IP (VoIP) clients etc) running on it [11]. Thus, a NCP can be designed that impersonates the mobile device and applications running on it during its sleep periods [3], [4], [11]. Further, depending on the mobile device, the NCP will completely shut down the device or just its WNIC (especially in case of smart phones) during its idle periods to extend the battery life.

IV. NETWORK CONNECTIVITY PROXY

As the first step, our work focused on the design and implementation of NCP which is more challenging and has potential to provide higher energy savings compared to other schemes [11]. The NCP allows its client devices to sleep during idle periods and maintain the presence on their behalf. It maintains its client reachability (ARP requests, NetBIOS name resolution, DHCP periodic lease requests), manageability (answering control and management protocols e.g., ICMP and SNMP), application reachability (responding to new connections and packets addressed to transport-layer listening ports) and application state (sending TCP keep-alive messages and application specific heart-beat messages) [3].

![Figure 1: Traffic Scheduling Proxy.](image1)

![Figure 2: Data Compression Proxy.](image2)
NCP maintains sleeping client’s link layer, network layer and/or application layer presence [4].

Basically, NCP classifies the packets addressed to sleeping devices in following categories: (i) No response required, i.e., discard/ignore the packet (multicast, broadcast, port scans, bridging, routing, etc), (ii) Minimal response required, simple predictable answer without need of heavy computation, (iii) Delay response required, i.e., packets can be buffered and processed later without affecting the remote application/protocol (IM messages, emails etc), (iv) Wake-up required, if the packets requires the device resources, such as new TCP connection requests, SNMP requests, and so on.

A. Architectural designs of NCP

The first architectural choice is between invisible or cooperative proxying. Invisible proxies do not advertise themselves in the network and inspect all traffic to/from the covered hosts to take decisions. However, its main limitations include: (i) host’s power state verification, (ii) host’s network presence confirmation, (iii) what applications to be proxied? To cope with these problems, a cooperative proxy can be designed which also requires a piece of software on the network host to communicate such information with the NCP.

A second architectural choice is the location of NCP service. The possible alternatives are:

1. **Endpoint proxying:** It embeds proxy in the host’s OS, protocols or applications, which informs the remote entity about power state changes and sleep intervals. Each peer is thus responsible to maintain state information for the sleeping host.

2. **On-board proxying:** It embeds proxy on host’s NIC which shares same IP and MAC addresses as host. The main limitation here is the low NIC’s memory and computation power to implement a complete NCP service covering many applications and protocols.

3. **Switch/Router proxying:** It places the proxying functionality on an always powered up network device, switch or router. These devices have an internal general purpose CPU running an embedded OS like Linux or FreeBSD which eases the deploying of complex and sophisticated routines supporting network presence for high-level protocols and applications. This approach requires a communication protocol to exchange information between hosts and the proxy.

4. **Standalone proxying:** It considers proxy deployment in some host other than network equipment that lies somewhere in the network. This choice provides high memory and high computational power to implement complex routines for large number of applications and hosts. This is most critical choice, as the standalone device must consume far less power than the savings from average number of sleeping devices.

5. **Dual proxying:** Each architectural solution discussed so far brings some benefits but also has some drawbacks. The dual proxying allows two proxies to work together to strengthen their effectiveness and balance out their weaknesses. Possible architectural solutions would be on-board & switch/router proxying, on-board & standalone proxying, switch/router & standalone proxying. This approach can also deal with Future Internet issues like DTN, mobility and intermittent connectivity.

V. MOBILITY MANAGEMENT

Till now, the proxying concept has only been considered for static networks while ignoring the host’s mobility. This section addresses the mobility issues and proposes architectural solutions considering the NCP but the same considerations with slight modifications will also hold for other proxying schemes addressed in Section III. In principle, maintaining network presence for mobile hosts has the same implications as for any fixed device [3], but there are two additional matters: (i) Mobile devices may change their point of attachment when in sleep mode, thus affecting both NCP operation and active sessions, (ii) Mobile devices may move out of wireless range quickly and might not have time to start NCP operation.

A. Main issues in low-power states

Many aspects of mobility could involve a device in low power: the user can bring a different device with him or may wish to move its running multimedia application, the device can move across different networks, and so on. If a device moves to a different network, it shall change its network address to stay compliant with the Internet architecture. This leads to several implications: (i) Devices in sleep mode are not aware of movement and thus cannot update network configuration, (ii) The NCP has out-dated information about device location, thus cannot wake up the device if required, (iii) The suspended sessions with remote peers will be lost after device wake-up.

1) **Host location updates:** The capability of tracking the location of sleeping hosts (e.g., IP address) is one of the prerequisite of the NCP operation. The tracking and location information can be managed by another server and the proxy could query it when necessary. The mobile device’s position could be updated with two approaches. The device periodically wakes up, checks its current link and, if different from the previous one, gets a new address and registers this information with the NCP or another location server. However, to have up-to-date information, frequent wake-ups are necessary which may result in shorter battery lifetime. The second solution would be to integrate simple network detection features in NICs: that will trigger host’s wake up on link attaching/detaching operations and then the host takes care of registering the current location with NCP. The second solution is undoubtedly more efficient than the previous one, as the host is at most awake only when necessary.

2) **Mobility implications in different proxying architectures:** Movements of sleeping devices affects two network parameters: IP addresses and link-layer forwarding mechanisms. Depending on the architectural choice (addressed in Section IV), the NCP may be affected by none, one or both of these elements. In end-point proxying, the NCP on remote
Table 1: Summary of main features about mobility management.

<table>
<thead>
<tr>
<th>Endpoint</th>
<th>Wake-up</th>
<th>Incoming Traffic</th>
<th>NATing</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-board</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Stand-alone</td>
<td>Yes</td>
<td>Yes, new connections must use old address</td>
<td>Yes</td>
</tr>
<tr>
<td>Dual proxy</td>
<td>Yes</td>
<td>Yes, both new and old addresses</td>
<td>Yes</td>
</tr>
</tbody>
</table>

peer knows the current location if it gets updated by an agent working on the NIC of sleeping device. Thus, the NCP can wake up the device when needed; further, it can NAT between the current IP address and that used to set up current connections, thus hiding mobility to the above applications. Incoming connections cannot wake the device up, but this is a general limitation of this architecture. In case of on-board proxying, the proxy moves together with the device. This is undoubtedly an advantage for movement detection; moreover, there is no problem for waking the host. However, once the proxy’s address has changed (usually the same as the host) it cannot get anymore packets addressed to the sleeping host. New incoming connections can be seen if the new host’s location is made available to all other hosts, that is quite simple with dynamic DNS. In a stand-alone/router/switch architecture the NCP does not move, thus it can behave like an ‘anchor’ for traffic addressed to nomadic sleeping hosts. Once it gets the up-to-date location of its client, it can wake it up and manage incoming connections; further, when the host awakes, the NCP can translate between the new and old addresses for connections set up with a previous address. In a dual-proxy architecture, the most convenient choice is having an on-board proxy and a stand-alone proxy. This solution brings all benefits of on-board and stand-alone proxying: devices can be woken up, incoming connections can be managed at both proxies (answering both the old and new addresses) and connections can be NATted. A brief summary of these main features is addressed in Table 1.

Link layer forwarding mechanisms should have little or no impact on wake-up. Wake-up packets are broadcasted because the MAC address of sleeping host is likely to be expired in forwarding caches of network equipment. On-board proxy keeps the host’s address active; however, such proxy is likely to send traffic and thus updates the forwarding caches with the new location. Placing the NCP in LAN makes it a good ‘anchor’ for mobile hosts, both within the local network and in the Internet. However, all traffic addressed to mobile hosts should make use of their ‘home’ address (where proxy lies), otherwise NCP will not be able to catch traffic addressed to sleeping hosts. Placing the NCP in the Internet makes it an ideal solution for all mobile users that either do not have a home network or do not wish to deploy their own NCP. Thus a global Internet-based NCP located anywhere in the Internet could simplify mobility management; however, it comes with issues for traffic diversion and NAT/firewall traversal.

B. Architectures for mobility management

This section proposes possible architectural solutions for maintaining network presence in case of mobile sleeping hosts while keeping remote peers unaware of any issue concerning low-power management and mobility.

1) An architecture based on Mobile IP: This solution fits very well all scenarios where a mobile host most of the time lies in a home network. Exploiting the Mobile IP (MIP) [13] framework keeps all layers above the network one unaware of mobility issues, thus making this framework suitable for any service and application. Here, two architectural designs are considered: (i) Home Agent (HA) and the NCP running on different devices, (ii) HA is integrated into the NCP.

The NCP and the HA are located in the same network; no Foreign Agent (FA) is required to keep the architecture as general as possible and suitable for the current Internet. When the device is at home, the HA does nothing while the device and the NCP behave like in a usual LAN scenario. In particular, if the device is asleep, the proxy catches its traffic and acts on its behalf as shown in Fig. 3. When the device moves to a foreign network, it keeps its IP address and additionally it gets a local one; a tunnel is used to exchange packets with the HA. When the device is awake, the NCP does nothing while the HA and the mobile device operate according to MIP framework. When the device goes to sleep, the NCP starts catching traffic addressed to the mobile device and acting on its behalf. This scenario is depicted in Fig. 4.

The main matter in this architecture concerns what happens when the Mobile Node (MN) is away from home and is sleeping. In this case, both the HA and NCP should catch packets addressed to the MN, both of them will try to resolve the host’s IP addressed into their own MAC. To avoid such conflict, two approaches could be followed depending on whether the MIP behavior is changed or not to account for the presence of sleeping devices. In any case, the NCP takes the preeminence in standing for the host.

In first approach, the NCP announces the MN at home. The MIP operation is not affected by sleeping devices and the standard behavior is preserved. However, as the HA thinks the MN be at home, it tears down the tunnel and the NCP must become aware of the MN’s current position and all following movements in order to send wake-up messages if needed. There are some major drawbacks and limitations with this approach. According to the standard behavior, the MN would keep the tunnel with the HA up when asleep, and that would be inconsistent with the state at the HA. Further, the MN registers with the HA when awake, but notifies its position to the NCP while asleep. The MN should also share with the NCP its current status about route optimization, so that the NCP could divert traffic back to the Home Network. Finally, when the MN wakes up or is awakened, the HA has to be updated like the MN was gone away and has to set up again the tunnel; this could delay communication and may negatively impact the interactiveness with sleeping devices.

In second approach, the HA and the NCP cooperate for handling sleeping hosts in foreign networks. In this case, an extension is added to standard MIP to make it aware of the ‘sleep’ condition (that could be inferred by seeing the MN’s traffic or, better, it could be notified by either the NCP or the MN). When this condition occurs, the HA stops catching packets for that MN and keeps the tunnel alive for delivering
wake-up messages and it continues to update the node’s position. The NCP keeps working as usual, with no additional tasks; wake-up messages are sent locally. The HA is also responsible to detect wake-up messages and to tunnel them to the Care-of Address (CoA) of the sleeping host. When the MN comes out from sleeping mode and the NCP stops proxying it, it should be ready to get all packets from the HA’s tunnel.

The second approach seems better, although it requires slight modification to MIP behavior. Further, the MIP framework allows extensions for the standard signaling to be added as new features are introduced; supporting low-power devices could indeed be an interesting add-on for this protocol.

In second architectural design, the HA is integrated into the NCP. This would make operation even easier, as no conflicts arise between the two functions. The NCP would catch packets addressed to MNs both to maintain network presence and to forward them to their current position. Functional description is the same as for the previous design.

2) An architecture based on Virtual NAT: Virtual NAT (VNAT) [14] was mainly designed for end-to-end applications, namely the remote host is aware of mobility and work together with the local mobile device. The prerequisite for keeping remote peers unaware of mobility is making use of an anchor which is Home Network (HN) as in MIP and where the local device is looked for. Packets addressed to a MN are hence forwarded to its HN; when the MN is present, packets are delivered as usual; if it is asleep, the NCP covers it. When the host moves away from its HN, its VNAT Migration informs the peer entity on the NCP. The NCP then starts NATting packets to/from the MN, as shown in Fig. 5. In the basic end-to-end paradigm of VNAT, new connections should be set up by using the current IP address as virtual identifier, thus avoiding the NAT overhead as long as possible; however, the transparent proxy paradigm used in this design requires all packets to cross the NCP for mobility and sleep support. Thus, all new sessions (both incoming and outgoing connections) should use the Home Address (HoA) of the MN, as in the MIP framework.

In VNAT no tunneling is required for mobility management. That implies less overhead in packet headers, but may rise more issues for firewall and NAT traversal: with a tunnel, just a single connection need to cross the firewall while in VNAT approach each single connection should be allowed, and this is very unlikely in almost all scenarios. Waking remote hosts is just a matter of sending a Magic Packet to their current location. The presence of security intermediaries can be an issue for this task too. Most issues concerning VNAT does not apply when it is integrated in the NCP framework: in particular, the NCP does not move and applications always use the same HoA which is uniquely assigned to their host.

VI. CONCLUSIONS

This paper has addressed the possible proxying schemes that can be adopted in order to improve the battery life of mobile devices. It specifically focuses on the NCP which is
Figure 5: Proxying mobile hosts with VNAT. Packets from/to travelling hosts are NATted both at the NCP and at the client.

somehow more challenging and provide higher energy savings compared to other proxying schemes. Till now the proxying schemes have been addressed for fixed networks without taking into account the host mobility. The main objective of this paper is to discuss the NCP for mobile devices, point out issues and propose solutions especially related to host’s mobility in low power states. Running the proxy on-board of mobile hosts makes the NCP service available anywhere; the proxy can easily detect movements, can change network configuration and can wake the host to do the same. However, this architecture does not allow maintaining existing connections with other hosts. Therefore, a stand-alone proxy, a proxy running on network equipment and dual proxy approach are the most promising solutions for mobility management, although sleeping hosts must take care of updating their location while they move around. This paper proposed two proxy architectural designs based on MIP and VNAT in order to support the host’s mobility and also addressed the basic requirements, issues and challenges associated with each approach. Our proposed architectural designs consider transparent proxying without support from third parties and require anchors to be present at the home location of mobile nodes. We have considered both using tunneling and NATting to forward packets to the mobile node in foreign networks, and discussed that tunneling may be most effective for NAT/firewall traversal at those remote sites.

ACKNOWLEDGMENTS

This work was partially supported by the European Commission in the framework of the FP7 ECONET (low Energy COntumption NETworks) project (contract no. INFSO-ICT-258454).

REFERENCES