Digital Communications and Networks (xxxx) xxxx–xxxx

HOSTED BY

ELSEVIER

Contents lists available at ScienceDirect

Digital Communications and Networks

journal homepage: www.elsevier.com/locate/dcan

Design and implementation of Ad-Hoc collaborative proxying scheme for reducing network energy waste☆

Rafiullah Khan[a,*], Sarmad Ullah Khan[b]

[a] School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, ECIT/CSIT - Northern Ireland Science Park, Queen's Road, Queen's Island, Belfast BT3 9DT, United Kingdom

[b] Electrical Engineering Department, CECOS University of IT and Emerging Sciences, Phase-6, Hayatabad, Peshawar, Pakistan

A R T I C L E I N F O

Keywords:
Green networking
Energy efficiency
Smartphone
Home gateway proxy
Power measurement
Universal Plug & Play

A B S T R A C T

Network devices are equipped with low power states but they are rarely activated due to their inability of maintaining network connectivity. Recently, Network Connectivity Proxy (NCP) concept has been proposed in literature as an effective mechanism to exploit full potential of low power features on network devices by impersonating their virtual presence. However, the NCP concept faces several open issues and challenges especially related to proxying of TCP connections and majority of daily used proprietary closed-source applications. This paper presents a new approach for reducing network energy waste through intelligent collaboration among daily used devices (e.g., desktop computers, laptops, tablets, smartphones etc). It guarantees to run applications on only and only one user device that is under active use at that specific moment. To reduce energy waste and allow idle devices to sleep, our approach also takes benefit from a light-weight home gateway proxy with very basic practically realizable functionalities. The proposed system not just reduces energy waste of fixed devices but also enables mobile devices to significantly improve their battery life. Our developed software prototypes allow devices to autonomously and seamlessly collaborate with each other without requiring any configuration or user input. Further, this paper also presents the basic performance evaluation of developed prototypes in real networking environments.

1. Introduction

A significant portion of the energy consumed by network devices is usually wasted due to under-utilization or staying powered-up even when idle [1]. About 2% of the overall US electricity is consumed by computers according to the Environmental Protection Agency (EPA) [2]. Whereas, computers in office environment stay idle for more than 60% of the time but are still kept powered-up [3]. The main reason due to which people never switch off their computers is to stay 'Always Connected' for remote access, VoIP and Instant Messaging (IM) clients and other Internet based applications. The Advanced Configuration and Power Interface (ACPI) proposed several low power management states for computers but they are rarely used due to their inability to maintain always-connected status. Thus, huge energy savings are possible if computers can extensively use ACPI low power states without losing the always-connected status [4].

The Network Connectivity Proxy (NCP) concept is recently proposed in literature as an effective mechanism to reduce the network energy waste by allowing unattended idle devices to sleep without losing network presence [5,6]. The NCP concept ensures that devices sleep for the maximum possible time and it wakes them up only when their resources are required [7]. To achieve this objective, the NCP seamlessly impersonates link, network, transport and application layers presence on-behalf of sleeping devices [5,8,9]. Although the NCP concept is quite beneficial,its practical realization is very complex and challenging especially when concerning proxying of TCP connections and ever increasing number of Internet-based applications [10]. Different proxying techniques have been investigated in the literature; however, they are always limited to a specific protocol or open source application [11,12]. The two most common ways of proxying applications are application specific stubs [8] and freeze/resume features [5,13]. Application stub is the light-weight version of original application derived or re-written from its source code. But freeze/resume approach requires modification to application to include new features. Both application proxying strategies are not applicable for proprietary closed-source applications. It is important to note that majority of commonly used applications in day-to-day life are proprietary closed source. Till now, the NCP concept lacks the capability to proxy closed-

Peer review under responsibility of Chongqing University of Posts and Telecommunication.

* Corresponding author.
E-mail addresses: rafiullah.khan@qub.ac.uk (R. Khan), sarmad@cecos.edu.pk (S.U. Khan).

http://dx.doi.org/10.1016/j.dcan.2016.11.001
Received 24 May 2016; Received in revised form 4 September 2016; Accepted 8 November 2016

source applications.

Similar to desktop computers, the Internet-connected smartphones also became an important part of our daily life. Today, the smartphones are capable to run the same latest operating systems as available for the desktop computers including Microsoft Windows, Ubuntu etc. Thus, the smartphones are able to offer all the same features and run the same applications as the desktop computers. The energy savings in the case of smartphone has no significant importance as it is already a very low power device. However, reducing energy waste can help to significantly improve the battery life [14]. Similar to computers, the applications on smartphone also demand always-connected status. Thus, the applications such as viber, skype, facebook messenger, whatsapp, vonage and other VoIP and IM applications periodically transmit/receive presence or heartbeat messages over Internet. These applications not only consume the mobile data but also use phone resources such as CPU, memory and hardware components such as WiFi, 3G/4G or some built-in sensors. In short, the smartphones battery life can be significantly improved if they run applications only when desirable.

In our daily life, multiple devices run the same applications concurrently (especially the smartphone always stay connected). In this paper, we present a new approach that can be easily realized practically to achieve all the benefits of NCP. Our proposed approach uses intelligent collaboration among daily used fixed (e.g., desktop computers) and mobile devices (e.g., smartphones, tablets, laptops). Instead of proxying, it guarantees to run applications on one and only one user device at any given instant. Thus, the applications run on either a smartphone or tablet or laptop or desktop PC or any other device currently under active use. For example, if the user is using a desktop computer, it will run all applications while other devices will reduce energy waste by stopping the applications. When the user is not using desktop computer, it can enter into sleep mode and a smartphone will maintain the presence of applications. Our proposed approach also uses a light weight Home Gateway Proxy (HGP) but with very basic practically realizable features. The HGP impersonates basic networking protocols (i.e., ARP, PING) on behalf of sleeping desktop computers and wake them up whenever necessary (e.g., remote access connection request is received). The proposed approach not only reduces the entire network energy waste but also helps in improving significantly the battery life of smartphones by running the applications only for short periods when desirable. Our main design goal in this paper is to achieve autonomous and seamless communication among different user devices without requiring any user input or configurations. For this purpose, our developed prototypes are based on the flexible and reliable Universal Plug & Play (UPnP) architecture. The UPnP architecture is suitable for any network device and guarantees zero configuration, auto-discovery and seamless communication among devices of interest. Finally, this paper also evaluates the performance of developed prototypes in real networking environments.

The rest of the paper is organized as follows: Section 2 presents a brief literature review. Section 3 motivates the proposed framework by addressing the issues and challenges in previous approaches. Section 4 presents detailed architectural design of our proposed system. Section 5 presents the light-weight HGP with very basic set of practically realizable features. Section 6 presents the implementation of software prototypes. Section 7 presents measurements and performance evaluation. Finally, Section 8 concludes the paper.

2. Background and related works

The NCP concept has recently been investigated by researchers in order to reduce the idle energy waste of desktop computers. The NCP impersonates link, network, transport and application layers presence on behalf of sleeping devices and wakes them up only when their resources are required. Although the NCP concept is quite beneficial, it still faces many open issues and challenges. Most of the literature works targeted some specific protocols or open source applications such as UPnP, Gnutella, Jabber clients and others [15,16,11,8].

The two main open issues in the NCP concept are proxying of TCP connections and proprietary closed-source applications. The initial attempt in preserving open TCP connections was based on introducing a new TCP header field ‘Connection Sleep’ [17]. This header field informs the remote peer about the power state transition of the device. Another approach is based on introducing a ‘shim’ layer between the application and socket interface [18]. This shim layer is responsible to freeze, start or stop a TCP session. However, both of the approaches require changes to standard TCP/IP operations. Another approach is based on using external Srelay SOCKS service [6] which relays every TCP connection between both peers. However, this approach faces scalability and performance issues as every connection is established through a third-party.

Agarwal et al. in [8,19] proposed the idea of using applications specific stubs for proxying of applications. However, writing stubs is a quite complicated process and is only possible for open source applications. A similar approach has been used for proxying of xChat and kaduChat application in [4]. Client devices use application-specific routines which continuously analyze the heartbeat messages of applications. Client devices transfer heartbeat information to NCP before sleeping. This approach is quite challenging if payloads are encrypted or varying in unpredictable fashion. Bolla et al. in [5,13] proposed a new approach using TCP session migration and introduced freeze and resume features in applications. The applications on client devices freeze their operations and migrate their TCP sessions to NCP before sleeping. Applications resume and TCP sessions migrate back from NCP when the client device wakes up. However, the approach is quite interesting but requires changes to applications to include freeze/resume features. The authors in [19] also proposed the idea of using virtual machines, where each virtual machine is responsible to maintain presence on behalf of a single sleeping device. However, this approach requires much memory and processing power which limit the scalability. Finally, to estimate the expected economic benefits, authors in [1,3] analyzed real network traffic traces for home and office environments and estimated the savings.

Similarly, for mobile devices such as smartphones, multiple complementary techniques have been proposed in order to improve their battery life. Most of the techniques are software based such as putting Wireless NIC (WNIC) in low power state during idle periods [20,21]. The authors in [22] proposed a stateful proxy called ‘scepter’ which is placed in the network infrastructure and reduces the number of control and data bits during transmission. Another approach is based on a transparent proxying for dynamic power aware scheduling [23]. The proxy maintains separate connection with both end peers and uses buffers to transmit data in bursts. Such approach for delay sensitive applications (such as youtube, Internet radio) and BitTorrent application has been addressed in [24,25]. A quite different approach of putting user interface instead of processor into sleep state has been proposed in [26]. It wakes up the user interface after processing (i.e., once the results are available). The concept similar to NCP has also been addressed for mobile device [14,27]. The authors have addressed challenges in design and implementation, however, the open issue of NCP still remains unresolved i.e., proxying of TCP sessions and closed-source applications.

Many smartphone applications on Android store (such as Go Power Master, Battery Doctor, Juice Defender, DU Battery Saver, PowerSaver etc) implement some interesting strategies. Stopping background running applications is the most common strategy which is based on some pre-defined conditions such as screen timeout, screen lock, specified battery threshold or specific time periods. Another interesting approach is turning OFF Internet connectivity/hardware components (such as WiFi, 3G/4G, Bluetooth etc) during undesirable periods. Once again, the Internet connectivity can be linked with screen timeout, battery threshold, screen lock, pre-specified time periods or periodi-