Department of Computer Science
Series of Publications A
Report A-2015-6

# Collaborative Traffic Offloading for Mobile Systems

Yi Ding

*To be presented, with the permission of the Faculty of Science of the University of Helsinki, for public examination in Hall 5, University Main Building, on 26th November 2015, at 12 o'clock noon.*

University of Helsinki
Finland

**Supervisors**
Sasu Tarkoma, University of Helsinki, Finland
Markku Kojo, University of Helsinki, Finland

**Pre-examiners**
Polly Huang, National Taiwan University, Taiwan, R.O.C.
Mika Ylianttila, University of Oulu, Finland

**Opponent**
Steve Uhlig, Queen Mary University of London, United Kingdom

**Custos**
Sasu Tarkoma, University of Helsinki, Finland

**Contact information**

Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Email address: info@cs.helsinki.fi
URL: http://www.cs.helsinki.fi/
Telephone: +358 2941 911, telefax: +358 2941 51120

Copyright © 2015 Yi Ding
ISSN 1238-8645
ISBN 978-951-51-1761-8 (paperback)
ISBN 978-951-51-1762-5 (PDF)
Computing Reviews (1998) Classification: C.2.1, C.2.2
Helsinki 2015
Unigrafia

# Collaborative Traffic Offloading for Mobile Systems

Yi Ding

Department of Computer Science
P.O. Box 68, FI-00014 University of Helsinki, Finland
Yi.Ding@cs.helsinki.fi
http://www.cs.helsinki.fi/u/yding/

PhD Thesis, Series of Publications A, Report A-2015-6
Helsinki, November 2015, 223 pages
ISSN 1238-8645
ISBN 978-951-51-1761-8 (paperback)
ISBN 978-951-51-1762-5 (PDF)

## Abstract

Due to the popularity of smartphones and mobile streaming services, the growth of traffic volume in mobile networks is phenomenal. This leads to huge investment pressure on mobile operators' wireless access and core infrastructure, while the profits do not necessarily grow at the same pace. As a result, it is urgent to find a cost-effective solution that can scale to the ever increasing traffic volume generated by mobile systems. Among many visions, mobile traffic offloading is regarded as a promising mechanism by using complementary wireless communication technologies, such as WiFi, to offload data traffic away from the overloaded mobile networks. The current trend to equip mobile devices with an additional WiFi interface also supports this vision.

This dissertation presents a novel collaborative architecture for mobile traffic offloading that can efficiently utilize the context and resources from networks and end systems. The main contributions include a network-assisted offloading framework, a collaborative system design for energy-aware offloading, and a software-defined networking (SDN) based offloading platform. Our work is the first in this domain to integrate energy and context awareness into mobile traffic offloading from an architectural perspective. We have conducted extensive measurements on mobile systems to identify hidden issues of traffic offloading in the operational networks. We implement the offloading protocol in the Linux kernel and develop our energy-aware offloading framework in C++ and Java on commodity machines and smartphones. Our prototype systems for mobile traffic offloading have been tested in a live environment. The experimental results suggest that our collaborative architecture is feasible and provides reasonable improvement in terms of energy saving and offloading efficiency. We further adopt the programmable paradigm of SDN to enhance the extensibility and deployability of our proposals. We release the SDN-based platform under open-source licenses to encourage future collaboration with research community and standards developing organizations. As one of the pioneering work, our research stresses the importance of collaboration in mobile traffic offloading. The lessons learned from our protocol design, system development, and network experiments shed light on future research and development in this domain.

**Computing Reviews (1998) Categories and Subject Descriptors:**
C.2.1 Network Architecture and Design
C.2.2 Network Protocols

**General Terms:**
Design, Experimentation, Measurement, Performance, Standardization

**Additional Key Words and Phrases:**
Mobile Traffic Offloading, Network Assistance, Energy Awareness, Software-Defined Networking

# Acknowledgements

***"You should learn to be grateful, truly."*** *– At the end of this meaningful journey, I could not stop thinking what my research path would be like without the support from so many exceptional people along the way. In this regard, I cherish earnestly the opportunity to express my gratitude through this official, serious, and yet very personal part of my dissertation.*

*First of all, my sincere thanks must go to my PhD supervisors Sasu Tarkoma and Markku Kojo at the University of Helsinki, and Jon Crowcroft (external advisor) at the University of Cambridge. Ever since I started my PhD, Sasu has offered me great opportunities to pursuit my research interests through his projects and international connections. I am grateful for his trust, his invaluable guidance, his encouragement especially in difficult times, and for teaching me the importance of conducting high impact research. Markku has been guiding my work for more than eight years. To me, he is a strict advisor who holds uncompromising high standard on research while being incredibly patient and considerate. I thank Markku for always keeping his office door open for me whenever I need his advice. His critical feedback had an immeasurable impact on this dissertation.*

*As my external advisor, Jon is a true mentor who not only provided me excellent research topics, but also educated me how to discover significant new ones. Through his unique Cambridge-style advising, I gradually learned what it takes to be original, how to stay vigorous, and what does 'fun of research' essentially mean.*

*I am grateful for having four outstanding PhD mentors: Esko Ukkonen, Jyrki Kivinen, Jussi Kangasharju, and Jukka Manner. They have offered me precious suggestions over countless official and casual meetings. I particularly appreciate the spontaneous discussions during coffee and lunch breaks. Through those conversations, they 'privately' coached me how to survive in the graduate school and illuminated numerous unwritten rules in the academia.*

*I also wish to thank Steve Uhlig for being my opponent. I thank Polly*

*Huang and Mika Ylianttila for reviewing this dissertation and providing their constructive feedback.*

*I gratefully acknowledge the financial support from the graduate schools including FIGS, DoCS and Hecse. My special thanks shall go to the Academy of Finland and Tekes for funding the research projects I am involved in. I thank Nokia for generously sponsoring my research through the Nokia Foundation Scholarships. I also thank the Association for Computing Machinery (ACM), HIIT, and the University of Helsinki for providing the travel grants that facilitated my attendance of academic conferences.*

*I need to stress that the Department of Computer Science at the University of Helsinki is a wonderful place to conduct research. I would like to thank our administrative and IT staff for making everything run smoothly. My special thanks go to several great people at the department who have endorsed me in many different ways: Hannu Toivonen, Petri Myllymäki, Jukka Paakki, Tomi Männistö, Pirjo Moen, and Tiina Niklander. I am very proud to have amazing colleagues to work with: Ilpo Järvinen, Laila Daniel, Sourav Bhattacharya, Eemil Lagerspetz, Kai Zhao, Ella Peltonen, Seppo Hätönen, Yanhe Liu, Emad Nikkhouy, Lauri Suomalainen, and Ibbad Hafeez. Without you, my life at the department would not have been so joyful and productive. There is one special person I must deliver my gratitude. Kimmo Raatikainen was my advisor who brought me into the research world back in 2007. Although he could not see the final outcome, I am indebted to him for the opportunity that has reshaped my career.*

*Through research projects and visits, I had the privilege to collaborate with many distinguished researchers. My special thanks go to Henning Schulzrinne, Jon Crowcroft and Pan (Ben) Hui for hosting me at their institutes. I would like to thank my co-authors Jouni Korhonen, Teemu Savolainen, Jörg Ott, Bo Han, Zhen Cao, Yu Xiao, and Hannu Flinck for their insightful suggestions. I am very grateful to Xiaoming Fu for that special reference letter he wrote for me in the summer of 2009.*

*I would like to thank my friends who have been accompanying me all these years. I thank Stefan Schönauer and Jouni Kleemola for the enduring friendship. Xiang Gan, Junyou Shen, Miaoqing Tan, Hao Wang and Hantao Liu are my Chinese friends who have helped me in the past. Their friendship and support shielded me from so many troubles during the toughest time when I arrived in Finland. I thank Hantao Liu in particular for encouraging me to go abroad and pursue an academic career. I also appreciate the accompany of Kun Zhang, Lu Cheng, Hongyu Su, Fang Zhou, Mian Du, Liang Wang, Yonghao Li, and Peng Liu. You made my life in Finland full of smiles and sweet memories.*

*Last and foremost, I owe so much to my families, especially my mother Liping Huang and father Tao Ding. My families have given me so much more than I can ever ask for. Through their unconditional love, I learned how to love. I am so lucky to have my wife Ya Xu who has been supporting me so firmly. She lights up my life with her love, care and encouragement. I must confess, without such great love to back me up, I could not reach this far.*

*For that one last suggestion quoted at the beginning from my mother pertaining to true gratitude, this dissertation is dedicated to her for the deepest love surrounding me all the years. I know I may never learn to do it right, but I am doing my best, all the time.*

*Aaron Yi Ding*
*Helsinki, November 2015*

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This dissertation proposes a novel traffic offloading architecture that can collaboratively utilize available resources and context from networks and mobile systems to support efficient and energy-aware offloading. In this chapter, we motivate the work, describe the problem statement, illustrate the research methodology, summarize the contributions, and present the organization of the dissertation.

For clarity, we use the term mobile network/access in this dissertation to refer to cellular network/access. The term mobile systems refers to hand-held computing devices that run dedicated operating systems.

## 1.1 Motivation

The impact of mobile and wireless communication is profound. Nowadays, mobile devices such as smartphones and tablets have become critical tools that offer great assistance and convenience to our daily lives. The widely deployed mobile network infrastructure has enabled pervasive connectivity to Internet services for mobile users. As indicated by the Cisco index, the global mobile devices grew to 7 billion in 2013, in which smartphones accounted for 77 percent of the growth. By 2018 the total number of mobile devices will grow to more than 10 billion [1].

Meanwhile, the combination of capacity increase in mobile access such as 4G/LTE and the popularity of smartphones has gradually changed the diurnal behavior of mobile users. The trend is towards always-on mobile applications that frequently access multimedia content such as video and audio. Owing to the success of mobile streaming and online social services, the global data traffic generated by mobile devices grew 81 percent in 2013 and will increase 11-fold between 2013 and 2018 [1, 2]. Such exponential rise of traffic volume leads to huge investment pressure on mobile operators' wireless access and core infrastructure, while the profits do not necessarily grow at the same pace. It is therefore urgent to find a cost-effective solution that can scale to the ever increasing number of mobile users and their demand for network capacity.

Among many visions, mobile network operators have identified traffic offloading as a viable solution to offload bulk Internet traffic to alternative access technologies such as WiFi. The trend to equip mobile devices with an additional WiFi interface and the fast growth of WiFi hotspot deployment[1] [2] also support this vision. Recent studies advocate that traffic offloading is feasible to relieve the investment pressure on the incumbent mobile network operators and improve customer experience in a cost-effective manner [3, 4]. The increasing number of operator-deployed WiFi access points to offload mobile data traffic [3] [4] [5] [6] further confirm the acceptance of this approach on a strategy level by the mobile network operators and Internet service providers (ISPs).

However, although recent research shows promising results on mobile traffic offloading [5, 6, 7, 8], there are still challenges remaining to be addressed. For mobile systems in particular, one of the key challenges is the impact of traffic offloading on energy consumption. As battery technology has developed slowly compared to the fast development of hardware and applications on modern mobile systems, battery life has become a critical bottleneck for user experience. Given the prior studies focused mainly on the operator perspective, such as how to maximize the amount of traffic offloaded, the user perspective such as energy consumption and service experience also deserves a thorough examination.

This dissertation investigates mobile traffic offloading to uncover its impact on mobile network operators and end users. We advocate that energy awareness and collaboration between network operators and end users are essential for mobile traffic offloading to benefit both sides in terms of efficiency and energy consumption. Therefore, we aim at a collaborative architecture that can better utilize the resources from networks and mobile systems to achieve efficient and energy-aware traffic offloading.

[1] Global number of public hotspots from 2009 to 2015: `http://www.statista.com/statistics/218596/global-number-of-public-hotspots-since-2009/`

[2] Global number of private hotspots from 2009 to 2015: `http://www.statista.com/statistics/218601/global-number-of-private-hotspots-since-2009/`

[3] AT&T Wi-Fi Internet Service for Home, Work & Mobile Network: `http://about.att.com/mediakit/wifi`

[4] Comcast XFINITY WiFi - Wireless Internet on the Go: `https://wifi.comcast.com`

[5] Cable WiFi Internet Access: `http://www.cablewifi.com/`

[6] Sonera Trustive Sim WiFi: `http://www.soneraglobal.com/trustive_wifi.php`

## 1.2 Problem Statement

In the context of mobile traffic offloading, there are two major perspectives to evaluate the impact: 1) the operator perspective and 2) the user perspective [3, 13]. The operator perspective concerns the technical and business values of incumbent mobile network operators such as Quality of Service (QoS) for customers, capital expenditure (CAPEX), and operational expenditure (OPEX). One of the major metrics for the operator perspective is the offloading efficiency [6], which is defined as the ratio of bytes offloaded to other channels, such as WiFi or femtocells [14], against the total bytes generated by the mobile systems. The user perspective concerns mainly the core values of end users, such as performance, service experience, and energy consumption. For the user perspective, the metrics include the key performance indicators (KPIs) that depict the Quality of Experience (QoE) such as throughput, latency, and in particular the battery life on mobile systems.

Because the initial drive of mobile traffic offloading originates from the mobile network operators that suffered from the radical increase of traffic volume in their radio access and infrastructure, early studies concentrate mainly on the operator perspective, such as how to maximize the traffic volume offloaded away from mobile access [5, 6, 7, 8]. Meanwhile, a recent study [15] has cast doubts on the existing schemes that overlook the user perspective and stresses that such negligence may cloud the future of mobile traffic offloading. As illustrated in [16], several user-centric factors can affect the outcome of mobile traffic offloading in terms of energy consumption and offloading efficiency. The key factors include hardware limitations, diversity of mobile systems, mobility, and variance of usage patterns. These observations signify that we must consider the user perspective in solution design. Furthermore, the success of an offloading mechanism can be judged by its adoption and deployment in the operational networks. In this respect, the open standard mechanisms, comparing to research proposals, have the advantage in deployment owing to the openness and extensibility. However, the standardization process may demand much more effort and is often time-consuming [17, 18, 26].

By observing the challenging issues in this domain, we seek answers to the following research questions:

1. What are the impacts of mobile traffic offloading? What are the implications for the mobile users and network operators? Why are existing proposals inadequate for the evolving networking environment? Which components are still missing?

2. Why is energy awareness important to mobile traffic offloading? How do we achieve energy awareness? How do we utilize the available resources and benefit from the collaborative assistance from network operators and end users?

3. How do we enhance the deployability and extensibility of a traffic offloading solution? How do we transfer a static and closed design to a dynamic and open one? Can the open standard software-defined networking (SDN) help in solving the problem? How do we integrate the SDN paradigm to mobile traffic offloading? What are the benefits and overhead of an SDN-based offloading platform?

## 1.3 Methodology

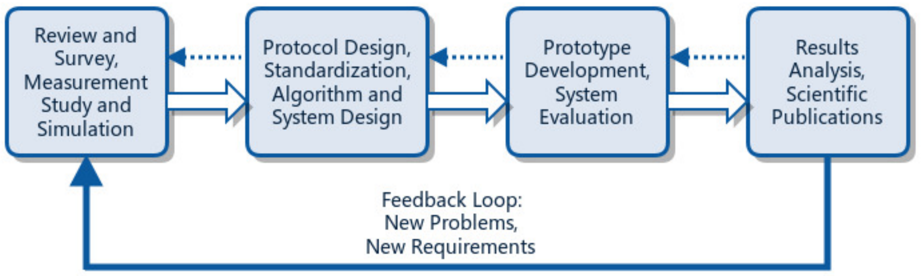

Review and Survey, Measurement Study and Simulation
Protocol Design, Standardization, Algorithm and System Design
Prototype Development, System Evaluation
Results Analysis, Scientific Publications
Feedback Loop: New Problems, New Requirements

Figure 1.1: Research methodology outline.

As illustrated in Figure 1.1, the research style of the author can be highlighted as ***measurement-based***, ***prototype-driven***, and ***systems-centered***. This research style drives our work and guides the research methodology that depends critically on four steps:

1. First, we conduct extensive literature studies, measurements, and initial simulations for the existing schemes and systems, in order to identify the major characteristics and limitations.

2. Second, we design protocols and system solutions based on the observations from the first step. To distinguish our work from the state of the art, we conduct a thorough analysis and compare our design against the existing schemes.

3. Third, we implement the most attractive design on prototype systems to uncover the complexity in a live environment, and evaluate its performance in operational networks.

4. Finally, we analyze the results, summarize the observations and share our contributions with research communities through scientific publications and standardization documents.

The research can be viewed as a progressive and iterative process. As a result, the feedback loop is crucial in our research development. When we encounter unexpected problems in design, prototype development or results analysis, we return to the previous step to re-investigate and refine our solutions. In particular, during the process of evaluating our proposal and analyzing test results, we often spot new issues and new challenges. Such insights motivate us to revisit the existing proposal and inspire us with new ideas. We take the new ideas and carefully apply them in the next round of research development. Such a feedback loop helps us to upgrade our design and step by step elevate it to a mature state.

## 1.4 Research History and Contributions

A large part of the results presented in this dissertation have been published in international conferences, workshops and as journal articles. The contributions were produced through a series of joint projects and research visits with noticeable impact on both research and standardization communities.

- The first part of the work was carried out at the Department of Computer Science, University of Helsinki through the 2-year Wireless Broadband Access (WiBrA) project in collaboration with Nokia, Nokia Siemens Networks and TeliaSonera [7]. Principal Investigator Markku Kojo was leading this project from 2010 to 2012. The outcome of the WiBrA project include a number of scientific publications and standard contributions [8]. The research in the WiBrA project forms the core of the Network-Assisted Offloading (NAO) framework (see Chapter 3). It is worth noting that in early 2011 when NAO was demonstrated during the Internet Engineering Task Force (IETF) meeting, being able to demonstrate selective IPv6-based network controlled offloading between WLAN and a live mobile network, even

[7] Wireless Broadband Access Project: `http://www.cs.helsinki.fi/group/wibra/`

[8] Wireless Broadband Access Project Publications: `http://www.cs.helsinki.fi/group/wibra/publications.html`

while roaming, was not possible for many IETF participants or even 3GPP delegates. Our NAO proposal is one of the pioneering works in the domain of IP traffic offloading.

- The second part of the contributions in this dissertation was produced in the Metropolitan Advanced Delivery Network (MADNet) project led by Dr. Pan Hui. It is a joint work of Deutsche Telekom T-Labs, University of Helsinki, and University of Maryland, College Park. The author was the key researcher from the University of Helsinki participating in this project from 2012 to 2013 together with Professor Sasu Tarkoma and Principal Investigator Markku Kojo. During the project, the author was based in University of Helsinki and made a research visit to Deutsche Telekom T-Labs in Berlin, Germany during Autumn 2012. The close collaboration with researchers at T-Labs boosted the progress and led to successful publications [16, 27]. The outcome of this joint project is the MADNet architecture for energy-aware traffic offloading (see Chapter 4).

- The third part of the work was initiated from a research visit at the University of Cambridge, Computer Laboratory, hosted by Professor Jon Crowcroft from June to December in 2013. The EIT-SDN project [9] led by Professor Sasu Tarkoma at the University of Helsinki provided research funding for this joint work. The outcome of our collaboration with Cambridge researchers is the SoftOffload design (see Chapter 5). A highlight of this visit is the joint work that summarizes our lessons and suggestions to bridge the gap between networking research and standardization [17, 18]. For the unique contribution to both research and standardization communities, this work was nominated as the ACM SIGCOMM "Best of CCR" editorial and selected for presentation in the ACM SIGCOMM conference in 2014 [10]. The implementation of the SoftOffload platform was conducted at the Department of Computer Science, University of Helsinki. The author also made a research visit to Columbia University in Autumn 2014, hosted by Professor Henning Schulzrinne. The development of SoftOffload benefited from our collaboration with the Columbia researchers.

[9] European Institute of Innovation and Technology (EIT) ICT Labs SDN Project: `http://www.cs.helsinki.fi/group/eit-sdn/`

[10] ACM SIGCOMM 2014 Conference Program: `http://conferences.sigcomm.org/sigcomm/2014/program.php`

Table 1.1: Overview of contributions.

| **Research Ques.** | **Methodology** | **Contributions** |
|---|---|---|
| What are the key issues & missing parts | Survey Study, Measurement & Simulation | Traffic offloading and energy awareness [13] (Section 2.2, 2.3); Study of competing data traffic in mobile access [20, 21, 22] (Section 3.1) |
| How to achieve collaborative & energy-aware offloading | System Design, Standardization, Protocol Design, Prototype Dev., Evaluation & Comparison | NAO framework [23], IPv4 offloading protocol [25] and solution comparison [26] (Section 3.2, 3.3, 3.4); MADNet energy-aware traffic offloading framework [27] and system evaluation [16] (Section 4.3, 4.4) |
| How to enhance deployability & extensibility of the design | Design Revisit, System Design, Platform Dev. & Evaluation | Review SDN design and revisit traffic offloading solutions [32, 33, 34] (Section 2.4, 5.2); SoftOffload platform design and implementation [28]–[31] (Section 5.3, 5.4) |

#### ***Research Contributions***

To the best of our knowledge, our work is the first to integrate energy and context awareness into mobile traffic offloading from an architectural perspective. As highlighted in Table 1.1, the main contributions of this dissertation include:

1. The author conducted a literature review and compared existing schemes to uncover the essential but yet missing components for mobile traffic offloading (discussed in Section 2.2, 2.3, and 2.4). The findings were published as a book section [13] in "*Smartphone Energy Consumption: Modeling and Optimization*" [19].

   To explore the key issues in mobile traffic offloading, the author participated in a series of experimental measurements and simulation studies to understand the characteristics of competing traffic, particularly the TCP-based web traffic against the UDP-based audio traffic, in a mobile access environment (discussed in Section 3.1). Ilpo

Järvinen is the main contributor to these experimental and simulation studies. The test results and analysis were published in conference papers [20, 21] and a technical report [22].

Since the survey and measurement studies provide valuable input for the design of mobile traffic offloading, this part of the work addresses the research question, "*What are the key issues and missing components for mobile traffic offloading?*"

2. The author was the key contributor to the Network-Assisted Offloading (NAO) framework [23] that aims to improve the efficiency of mobile traffic offloading (discussed in Section 3.2). By collaborating with Dr. Jouni Korhonen and M.Sc. Teemu Savolainen, the author co-designed the IPv4 offloading protocol [25] (discussed in Section 3.2.3) for NAO framework and pushed the proposal into the standardization process at the Internet Engineering Task Force (IETF) [11] to extend its industrial impact. The author was the architect of NAO and led the prototype implementation of the proposed IPv4 offloading protocol in the Linux kernel. It was the author who set up the testing environment in the university laboratory and conducted live experiments in an operational network (discussed in Section 3.3.3). Together with Jouni Korhonen and Teemu Savolainen, the author analyzed the standard solutions and compared them with NAO (discussed in Section 3.4). The testing results and comparison study were published in a journal article [26].

   The author is one of the key contributors in collaboration with Dr. Pan Hui, Dr. Bo Han, and Dr. Yu Xiao to the energy-aware offloading architecture (MADNet) that can collaboratively utilize available resources and context from networks and end systems to enable energy-aware WiFi offloading [27]. The design is driven by the measurement studies conducted by Dr. Bo Han and Dr. Yu Xiao towards metropolitan WiFi access and smartphone energy consumption (described in Section 4.2). The author enhanced and integrated the energy-aware offloading algorithm, a WiFi-based localization scheme and a mobility prediction design dedicated to the MADNet architecture (discussed in Section 4.3). To evaluate MADNet and demonstrate its effectiveness, it was the author who developed the prototype in C++ on both smartphones and commodity servers, set up the testing environment, and conducted evaluations in a live network environment using the prototype system (discussed in Section 4.4).

[11] Internet Engineering Task Force (IETF): `https://www.ietf.org/`

The measurement analysis and experiment results were published in a conference paper [16].

The work on NAO and MADNet addresses the research question, "*How do we achieve collaborative and energy-aware mobile traffic offloading?*"

3. The author further proposed a programmable SDN-based offloading platform (SoftOffload) [28]–[31]. The design of SoftOffload is based on our survey studies on SDN together with Professor Jon Crowcroft, Hannu Flinck and colleagues at the University of Helsinki [32, 33, 34]. The author conducted measurement studies in cellular and WiFi networks with Yanhe Liu (discussed in Section 5.2). It was the author who designed the SoftOffload system architecture for enabling collaborative offloading (discussed in Section 5.3). The author co-designed the context-aware algorithm, implemented the system prototype, built the testbed and conducted experiments together with Yanhe Liu. The implementation and experimental results showed that SoftOffload can improve the deployability and extensibility by leveraging the openness and programmability of SDN (discussed in Section 5.4).

   This part of the work addresses the research question, "*How do we enhance deployability and extensibility, and what are the lessons?*"

The development of this dissertation follows the research methodology illustrated in Figure 1.1. In each project phase, we first conduct a thorough literature review and a set of measurements/simulations (Section 3.1, 4.2 and 5.2). Based on the observations from the initial stage, we propose and design a dedicated protocol extension (Section 3.2), framework, algorithm (Section 4.3) and platform (Section 5.3). By building the prototype systems, we evaluate our proposals in operational networks (Section 3.3, 4.4 and 5.4). At the end of each phase, we analyze our evaluation results and reflect on the findings (Section 3.4, 4.5 and 5.5). The analytic process in our research development helps reveal hidden issues and new requirements. Furthermore, the accumulated insights enable us to inspect and refine our design. By driving our research in a progressive and iterative manner, we achieve the final collaborative architecture for energy-aware traffic offloading.

## 1.5 Structure of the Dissertation

The rest of this dissertation is organized as follows. Chapter 2 describes the background for this dissertation. It covers traffic offloading, energy-aware design for mobile systems, and software-defined networking (SDN) in mobile access. Chapter 3 introduces the Network-Assisted Offloading (NAO) framework, covering the protocol design, framework implementation, and experiments. Chapter 4 presents the architecture for collaborative and energy-aware traffic offloading. It describes our measurement findings in metropolitan WiFi networks, design principles, system implementation and evaluation. Chapter 5 presents the SoftOffload platform, covering our measurements on WiFi offloading performance and energy consumption, platform implementation, and experimental evaluation. Finally, Chapter 6 concludes this dissertation by summarizing our contributions, discussing open issues, and outlining future work.

# Chapter 2

# Traffic Offloading in Mobile Networks

This chapter presents the background of this dissertation. First, Section 2.1 provides an overview and discusses the challenges and opportunities in the evolving mobile and wireless networking environment. Section 2.2 provides an overview of traffic offloading for mobile systems, focusing on the latest development of research and standardization. Section 2.3 presents the state of the art in collaborative and energy-aware design dedicated for mobile systems. Section 2.4 introduces the latest development of software-defined networking (SDN) for mobile networks. Finally, Section 2.5 summarizes the chapter.

## 2.1 Background

The development of wireless communications and networking technologies have expanded the possibilities of what mobile networks can offer to achieve pervasive and mobile Internet access. In this section, we provide an overview of the current mobile and wireless networking environment as the technical basis for this dissertation. We emphasize on the system architecture and characteristics of access technologies that influence mobile traffic offloading.

### 2.1.1 Mobile networking environment

The current mobile and wireless networking environment consists of heterogeneous access technologies and network architectures. Based on the geographical coverage, the existing technologies can be categorized into two classes: wireless wide area network (WWAN) and wireless local area network (WLAN).

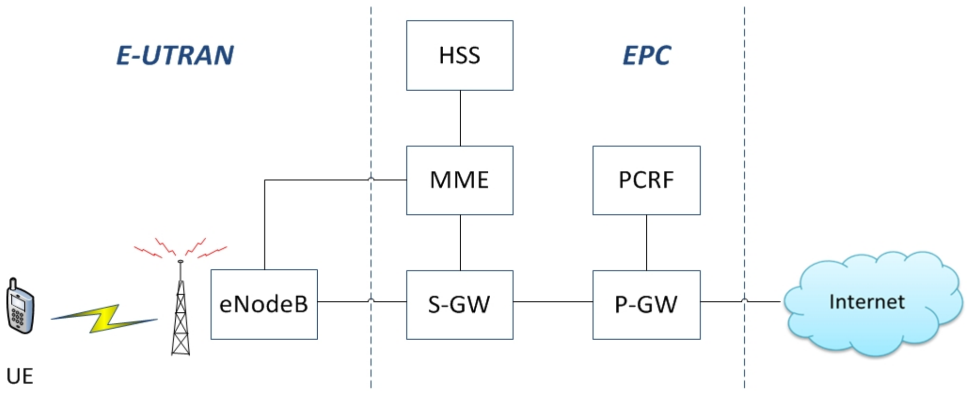


Figure 2.1: LTE system architecture.

#### *Wireless Wide Area Networks*

In the wireless wide area network (WWAN) domain, the 3rd Generation Partnership Project (3GPP) [1] is the dominant standardization group that standardizes access technologies and network architectures for terrestrial mobile networks. Based on the Institute of Electrical and Electronics Engineers (IEEE) 802.16 standard, the Worldwide Interoperability for Microwave Access (WiMAX) Forum [2] also proposed the Mobile WiMAX [35, 36] for WWAN.

For the fourth generation of mobile telecommunications technology (4G) [37], 3GPP standardized the Long Term Evolution Advanced (LTE-Advanced) in the Release 10 specification [3]. By applying advanced carrier aggregation and multi-antenna techniques, the theoretical peak rates of LTE-Advanced can reach 3 Gbit/s in downlink transmissions (from access network to mobile device) and 1.5 Gbit/s for uplink (from mobile device to access network) over the wireless link. The Mobile WiMAX specified in IEEE 802.16m [38] also meets the 4G requirements by supporting the peak rate of 1 Gbit/s in stationary downlink transmissions.

Prior to 4G, 3GPP standardized Long Term Evolution (LTE) in Release 8 [4]. As illustrated in Figure 2.1, the LTE architecture includes Evolved Universal Terrestrial Radio Access Network (E-UTRAN) [39] for the access network and Evolved Packet Core (EPC) [40] for the core network. The theoretical peak rates of LTE can reach 300 Mbit/s in downlink transmis-

[1] The 3rd Generation Partnership Project (3GPP): `http://www.3gpp.org/`
[2] WiMAX Forum: `http://www.wimaxforum.org/`
[3] 3GPP Rel. 10: `http://www.3gpp.org/specifications/releases/70-release-10`
[4] 3GPP Rel. 8: `http://www.3gpp.org/specifications/releases/72-release-8`

sions and 75 Mbit/s in uplink transmissions.

The core components of E-UTRAN and EPC include the evolved NodeB (eNodeB), Serving Gateway (S-GW), Packet Data Network Gateway (P-GW), Mobility Management Entity (MME), Home Subscriber Server (HSS), and Policy Control and Charging Rules Function (PCRF).

In E-UTRAN, eNodeB is the central entity that connects the user equipment (UE) to the EPC. Besides providing radio resource management (RRM) functions for UE, eNodeBs are interconnected with each other and connected to EPC. Since LTE integrates the radio controller function into the eNodeB, it enables a tight interaction between different protocol layers in the access network and thus improving efficiency.

In Evolved Packet Core (EPC), Serving Gateway (S-GW) transfers IP packets and serves as the local mobility anchor when the UE moves between eNodeBs. An IP traffic flow in EPC is referred to as a “bearer” which bears a defined QoS between the gateway and the UE. S-GW retains the information about bearers and collect charging details. To support interworking, S-GW also serves as the mobility anchor for other 3GPP technologies such as general packet radio service (GPRS). The Packet Data Network Gateway (P-GW) in EPC is the point of interconnect between EPC and the external IP networks, and responsible for IP address allocation for UEs. It enforces QoS by filtering IP packets into different QoS-based bearers based on the Traffic Flow Templates (TFTs). P-GW also serves as the mobility anchor for interworking with different non-3GPP technologies such as WLAN and WiMAX. The Mobility Management Entity (MME) is the control node in EPC that processes the signaling related to mobility and security for E-UTRAN by utilizing the Non Access Stratum (NAS) protocol [41]. The main functions supported by MME include bearer management and connection management. The bearer management covers the establishment, maintenance and release of the bearers. The connection management includes the establishment of the connection and security setup between UE and the network. The Home Subscriber Server (HSS) in EPC contains the subscription data for users including the QoS profiles and access restrictions for roaming. HSS also holds dynamic information such as the identity of MME to which a user is currently registered. The Policy Control and Charging Rules Function (PCRF) is responsible for policy control, decision making and controlling the flow-based charging functionalities in the Policy Control Enforcement Function (PCEF) which resides in the P-GW. The PCEF managed by PCRF provides the QoS authorization that ensures the data flows of a certain user is treated in accordance with the user’s subscription profile. Accompanied by the System Architecture Evolution (SAE) that aims to develop EPC for the core network, E-UTRAN and EPC together form the Evolved Packet System (EPS).

For the third generation of mobile telecommunications technology (3G) [42], 3GPP first introduced the Universal Mobile Telecommunications System (UMTS) in Release 99 [5], which was originally based on Wideband Code Division Multiplexing (WCDMA) with a transfer rate of 384 kbit/s. 3GPP further improved WCDMA through the High Speed Packet Access (HSPA) in Release 5 [6] and Release 6[7] and the Evolved High Speed Packet Access (HSPA+) [8]. According to 3GPP Release 11, the theoretical peak rates of HSPA+ can reach 672 Mbit/s in downlink transmissions and 168 Mbit/s in uplink transmissions [43, 44]. Besides 3GPP-based systems, the 3rd Generation Partnership Project 2 (3GPP2) [9] also standardized CDMA2000 by using Code Division Multiple Access (CDMA) [45] as the access method. Based on Evolution-Data Optimized (1xEV-DO), CDMA2000 can support 14.7 Mbit/s in downlink transmissions and 5.4 Mbit/s in uplink transmissions [46, 47].

As an upgrade from the globally deployed 3GPP Global System for Mobile Communications (GSM), the UMTS network architecture consists of UMTS Terrestrial Radio Access Network (UTRAN) and Core Network (CN) as shown in Figure 2.2. By supporting the UMTS-based access technologies such as WCDMA, HSPA, and HSPA+, UTRAN provides connectivity between the UE and the UMTS Core Network. The UMTS CN consists of a circuit-switched core network for traditional GSM-based services and a packet switched core network allowing IP access to the Internet.

As illustrated in Figure 2.2, the core components of the UMTS system include NodeB, Radio Network Controllers (RNC), Serving GPRS Support Node (SGSN), Gateway GPRS Support Node (GGSN), Home Location Register (HLR), Authentication Center (AuC), and Mobile Switching Center (MSC).

In UMTS Terrestrial Radio Access Network (UTRAN), NodeB is responsible for wireless communications between the UE and the core network. The Radio Network Controller (RNC) is a governing entity responsible for controlling the NodeBs connected to it. The RNC carries out radio resource management and supports mobility management in the UTRAN.

In UMTS Core Network (CN), the Serving GPRS Support Node (SGSN) is responsible for the delivery of data packets from and to the UEs within its

[5] 3GPP R. 99: `http://www.3gpp.org/specifications/releases/77-release-1999`
[6] 3GPP Rel. 5: `http://www.3gpp.org/specifications/releases/75-release-5`
[7] 3GPP Rel. 6: `http://www.3gpp.org/specifications/releases/74-release-6`
[8] 3GPP Rel. 7: `http://www.3gpp.org/specifications/releases/73-release-7`
[9] The 3rd Generation Partnership Project 2 (3GPP2): `http://www.3gpp2.org/`

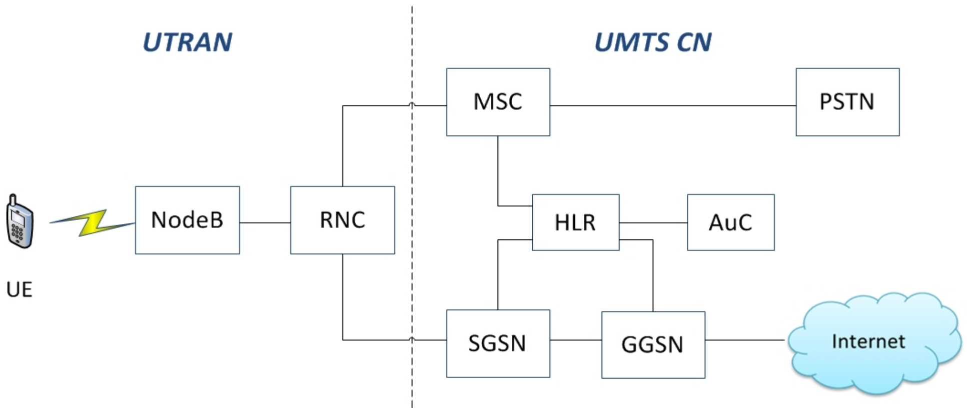


Figure 2.2: UMTS system architecture.

geographical service area. The main duties of SGSN include packet routing, mobility management, logical link management, authentication, and accounting. The Gateway GPRS Support Node (GGSN) is responsible for interconnecting the GPRS core network to the external networks such as Internet and X.25. Its main functions include IP address pool management, address mapping, and QoS enforcement. GGSN is the anchor point for the UE mobility in UMTS networks. The Home Location Register (HLR) is a central database that contains details of each authorized subscriber. The Authentication Center (AuC) is responsible for authenticating UEs that attempt to connect to the core network. For the UMTS circuit-switched core network, the Mobile Switching Center (MSC) is the primary service delivery node connecting to Public Switched Telephone Network (PSTN) that supports traditional GSM voice and messaging. It sets up and releases the end-to-end connection, handles mobility and handover requirements during the call, and takes care of charging and real-time pre-paid account monitoring.

#### *Wireless Local Area Networks*

The IEEE 802.11-based Wireless Local Area Network (WLAN) is well established as a convenient and low-cost technology to provide wireless broadband access within a small geographical area. Through the popular Wi-Fi branding, the Wi-Fi Alliance [10] has made the 802.11 WLAN well known to the public as the Wi-Fi (or WiFi) technology.

[10] Wi-Fi Alliance: `http://www.wi-fi.org/`

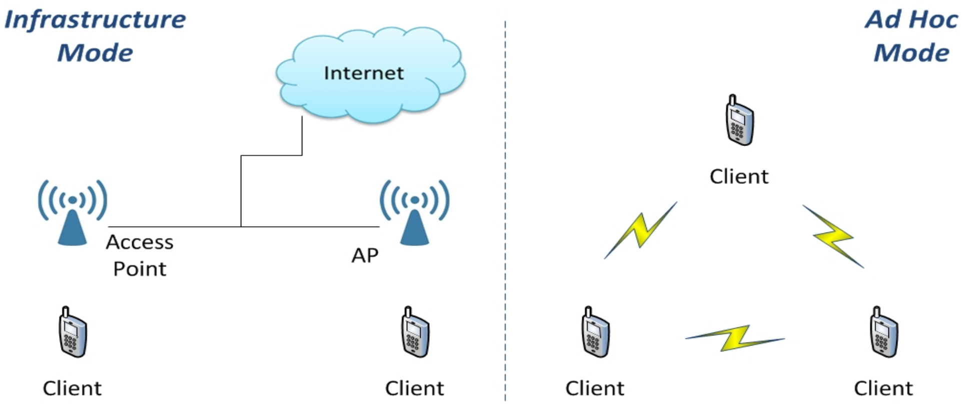


Figure 2.3: WLAN infrastructure and ad doc modes.

In the WLAN architecture, all entities that can connect to the network through a wireless medium are commonly referred to as stations. As shown in Figure 2.3, there are two types of WLAN stations: access point (AP) and client. A WLAN AP is typically a wireless router responsible for communicating with wireless enabled devices over the wireless link. A WLAN client can be a mobile device or fixed device equipped with a wireless network interface.

As illustrated in Figure 2.3, IEEE 802.11 WLAN has two basic operation modes: infrastructure mode and ad hoc mode. In infrastructure mode, clients communicate through the access points with each other and the external networks such as Internet. In ad hoc mode, clients communicate with each other in a peer-to-peer manner without the support of access points.

Since the first IEEE 802.11 standard released in 1997 [48], there are now several variants of 802.11 with enhancement to gain higher transmission rates. Being the first one widely deployed on university campuses, 802.11b [49] uses 2.4 GHz frequency band and can achieve a transmission rate of up to 11 Mbit/s. On the 5 GHz frequency band, 802.11a [50] is capable of transmitting at 54 Mbit/s. By applying advanced channel encoding schemes, 802.11g [51] uses the same 2.4 GHz frequency band as 802.11b and can achieve 54 Mbit/s in transmission. The newer 802.11n [52] standardized in 2012 can operate on both 2.4 GHz and 5 GHz. Its theoretical peak rate can reach 600 Mbit/s. The latest 802.11ac [53] uses 5 GHz frequency band and is capable of transmitting at 1 Gbit/s. To achieve a higher transmission rate, 802.11ad [54] uses 60 GHz frequency band and can achieve a theoretical peak rate of up to 7 Gbit/s. Although the initial deployment of 802.11 WLAN started from the university and enterprise environment, it has become a popular wireless broadband solution used in many public locations nowadays [11] [12] [13].

### *Heterogeneous Mobile Access Environment*

The concept of wireless overlay networks was first introduced in 1998 for the environment consisting of different access technologies varying in coverage, throughput, and mobility support. By definition, wireless overlay networks are formed by a hierarchical structure of room-size, building-size, and wide area data networks that can provide network connectivity to a large number of mobile users in an efficient and scalable way [55].

Owing to the fast development of wireless and mobile technologies, mobile networks resemble the wireless overlay networks by consisting of heterogeneous access technologies including LTE, UMTS, WiMAX, and WiFi. To deliver the best possible connectivity in such multi-access environments, 3GPP has proposed an integrated architecture via the System Architecture Evolution (SAE) [56, 57]. As illustrated in Figure 2.4, the architecture encompasses both 3GPP and non-3GPP access technologies and unifies multiple access networks through the all-IP core infrastructure.

For 3GPP accesses such as UTRAN and the legacy GERAN, SAE connects them through the UMTS/GPRS core to the EPC by using an interconnecting anchor. The latest E-UTRAN directly connects to the EPC towards Internet via the P-GW. For non-3GPP technologies, SAE introduces the Evolved Packet Data Gateway (ePDG) to enable interoperability between non-3GPP accesses and the 3GPP EPC. The main function of the ePDG is to secure the data transmission with a UE connected to the EPC over an untrusted non-3GPP access. For this purpose, the ePDG is responsible for authentication and acts as a termination node of security tunnels established with the UE.

To provide information to UEs about connectivity to 3GPP and non-3GPP access networks, SAE has introduced one important function, Access Network Discovery and Selection Function (ANDSF) [58]. The purpose of ANDSF is to assist UEs to discover the access networks in their vicinity and to provide policy rules to prioritize and manage connections to these networks. Since modern UEs support access to UMTS, LTE and WiFi by using the dedicated network interfaces, the SAE architecture provides

[11] PanOULU Open Wireless Internet Access: `https://www.panoulu.net/`

[12] Wi-Fi Hot Sports: NYC Parks: `http://www.nycgovparks.org/facilities/wifi`

[13] Helsinki Airport Free Wi-Fi: `http://www.finavia.fi/en/helsinki-airport/services/internet-and-working-stations/free-wifi/`

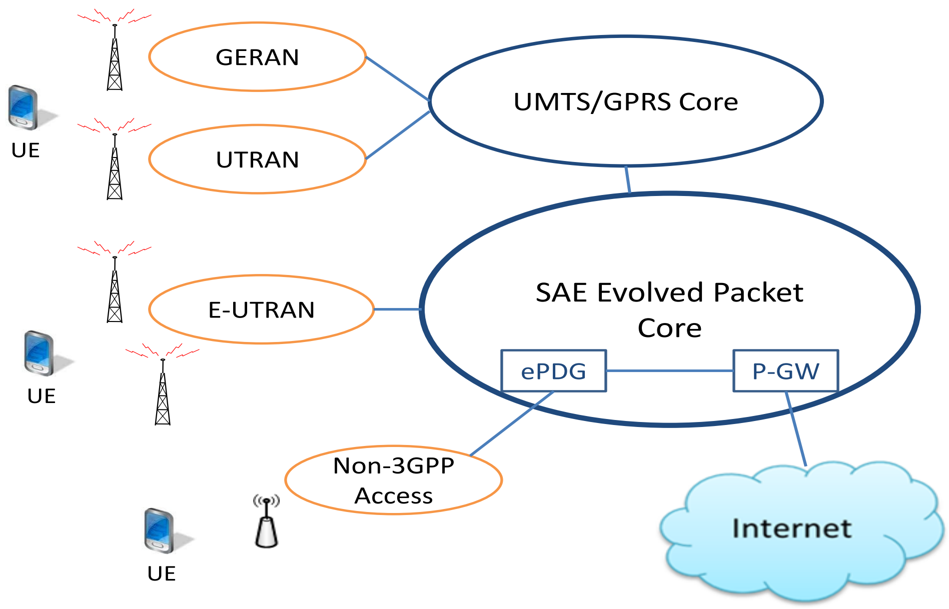


Figure 2.4: SAE multi-access architecture.

a foundation to deliver pervasive connectivity and manage mobile traffic across different access technologies and networks.

### 2.1.2 Challenges and opportunities

The phenomenal growth of mobile devices and the infrastructure upgrades from 3G to 4G/LTE have made mobile access a primary channel for more and more people to access Internet services. This fast pace of changes in the mobile access environment has accelerated the technical innovation and business growth, with good examples of mobile streaming and cloud services.

Meanwhile, with the inherent mobility support and pervasive coverage, the performance improvement in mobile access in terms of latency and throughput has brought two notable outcomes: 1) *mobile users nowadays utilize mobile access in a similar way as they use fixed access for all sorts of services such as online social networks and multimedia streaming*. 2) *mobile network operators experience an exponential traffic growth in their network infrastructure*. The shift of usage pattern and traffic surge have generated investment pressure for mobile network operators to balance expenditure and profits.

Furthermore, driven by the fast development of mobile services, device hardware and software, the mobile traffic volume will continue to grow. The projected global mobile data traffic will increase nearly 10 times between 2014 and 2019, and nearly three-fourths of the mobile data traffic will be video traffic by 2019 [1]. Along with the proliferation of mobile devices and services, the mobile access environment is advancing as well. Nowadays multiple wireless communication technologies in both licensed and unlicensed frequency bands have formed a heterogeneous access environment that offers end users flexible connectivity. However, this growing heterogeneity also requires mobile network operators to manage distinct access, backhaul and core networks, and increases costs and operational complexity.

#### ***Key Challenges***

The exponential growth of mobile traffic volume, the quickly evolving mobile services, and the inherent need for simultaneously operating over multiple wireless technologies impose significant challenges for mobile networks across technical and business perspectives. We highlight the key challenges as follows.

- *Difficulty to Scale* – With mobile traffic continuing to rise, the incumbent static over-provisioned networks are inflexible and too costly to keep up with the demand. The traditional approach of scaling network capacity with additional network equipment, such as by installing more base stations, is still available, but it is not cost-effective and viable considering the pace at which the demand is increasing.
- *Difficulty to Manage* – Existing mobile networks rely on legacy operations support systems and management systems that require significant expertise and platform resources to manage the network. Because these systems depend heavily on manual setup, networks are prone to misconfiguration, errors, and lengthy delays in provisioning and troubleshooting. It often takes weeks or even months to introduce new services because of the manually intensive processes for service activation, delivery, and assurance, which further complicates the solution deployment.
- *Growing Complexity* – Besides the growing heterogeneity in access technologies, the growing complexity of mobile networks has spanned multiple dimensions. It includes network architectures, mobile services, and network usage patterns. Because the operational environment consists of different technologies and operators, it leads to

complex negotiation processes, privacy concerns, and potential conflicting policy and QoS requirements. The traffic pattens generated by new services are often transient and unpredictable. Without novel approaches, the existing networks are inflexible to handle such dynamic and complex situations.

- *Demand for Interoperability* – Although mobile network operators are quickly upgrading their network infrastructure, the legacy telecommunication systems are still in use. This leads to demand for interoperability so that various technologies and networks can coexist and safeguard network reliability. All those create a challenge to network management in which a design should be backward compatible during the transition from older to newer technologies without impacting the customer experience.
- *Lack of User Considerations* – For profit and cost concerns, mobile network operators often prioritize the operator perspective and neglect the user perspective when it comes to solution design and deployment. This bias has a negative impact on the recent development for mobile traffic offloading schemes. The reality is that we have a lack of collaborative and balanced mechanisms that take into account both network and user perspectives.
- *Bottleneck for Innovation* – The legacy telecommunication systems deployed in mobile networks form a closed loop which lacks flexibility and extensibility. This creates a bottleneck limiting the rate of innovation for both design and adopting new solutions. Due to the ever-increasing complexity in mobile networks, such a bottleneck will burden the network operation and increase CAPEX and OPEX.

### ***Opportunities and Trends***

To address the challenges in mobile networks, we can benefit from emerging opportunities created by the recent development of the mobile industry across network infrastructure, end devices, research, and open standards.

First, the core infrastructure of mobile networks is becoming more inclusive to support diverse access technologies. Converged by the all-IP core, the 3GPP SAE architecture as shown in Figure 2.4 is a good example to integrate non-3GPP technologies into 3GPP networks. The fast adoption of cloud computing and network function virtualization (NFV) [59] techniques will allow mobile networks to further meet the demand for flexibility and scalability. On the network edge, different access technologies provide abundant choices for network providers to enrich their services based on the budget and user requirement. For instance, WiFi technology is favored by network operators as a popular wireless broadband solution to alleviate the pressure on the overloaded mobile access and experience a fast adoption in both public and private domains. The evolving mobile network infrastructure creates a foundation to support novel design for traffic offloading.

Second, mobile devices are becoming more powerful and versatile. Supported by powerful multi-core CPUs and abundant memory, modern devices such as smart phones and tablet devices are capable of handling demanding tasks. Multiple network interfaces such as cellular, WiFi, and Bluetooth [60] can allow end users to fully utilize network resources in the existing multi-access environment. The built-in sensors on mobile devices have been improved over the years and can provide rich context such as motion, location, and surroundings. Such user context has huge potential to make traffic offloading more active and collaborative.

Third, abundant research proposals have formed a comprehensive knowledge base to support mobile traffic offloading. The accumulated contributions cover IP mobility, network selection, and handoff management. To support seamless connectivity, handoff and mobility solutions have been studied [61, 62]. For handoff decision and network selection, there are studies exploring vertical handoff in multi-access environments [63, 64, 65, 66, 67]. To improve TCP performance, researchers have proposed a cross-layer approach to make TCP adaptive to wireless and mobile environments [68, 69].

Furthermore, the development of open standards also promotes extensibility and flexibility in mobile networks. For mobility support, IETF has standardized Mobile IP [70, 71] and Proxy Mobile IP [72, 73]. For handover management, IEEE proposed 802.21 that specifies media access-independent mechanisms to optimize handovers between heterogeneous IEEE 802 systems and between IEEE 802 systems and cellular systems [74]. The Open Networking Foundation [14] has promoted OpenFlow [75] as the standard communication interface to support software-defined networking (SDN) deployment.

In brief, the technical maturity and research development have paved the way to tackle the challenges in mobile networks. We highlight three topics that bring positive impact to this field: mobile traffic offloading, energy awareness, and software-defined networking (SDN).

[14] Open Networking Foundation: `https://www.opennetworking.org/`

## 2.2 Mobile Traffic Offloading

As mobile traffic volume continues to expand, network operators are seeking viable mechanisms to augment their network capacity to meet the ever-increasing demand. These mechanisms include upgrading the access networks to LTE/4G with better spectral efficiency, improving backhaul capacity, and offloading traffic from the mobile access to alternative channels such as WiFi and femtocells [14]. Being one of the techniques in network management, traffic offloading has been adopted by mobile network operators to alleviate the pressure on their networks overloaded by the surge of mobile traffic. For instance, owing to the advantage of low-cost and technical maturity, WiFi offloading has become a popular choice. In 2013, 45 percent of total global mobile data traffic was offloaded through WiFi or femtocell [1].

The key enabler for mobile traffic offloading is the fast development of wireless communication technologies. Nowadays, smartphones can support a rich set of communication technologies such as LTE, HSPA, WiFi, and Bluetooth. The new standards such as WiFi-direct [15] and Bluetooth Smart [88] further enhance the flexibility and energy efficiency for device-to-device communications. On the network side, mobile operators are upgrading their infrastructure to LTE and LTE-advanced for enhanced performance. WiFi and femtocells are gaining popularity in metropolitan areas to offer diverse and convenient wireless access.

Following the trend, 3GPP has put considerable effort in standardizing IP-based offloading solutions for the EPC with a tight integration to 3GPP network architecture [58, 112, 113]. Research communities also showed convincing results on the effectiveness of mobile traffic offloading through experimental studies [5, 6, 7, 8]. To cover the recent development, we illustrate the procedure of mobile traffic offloading and present key proposals from research community and standardization bodies.

### 2.2.1 Traffic offloading procedures

The typical mobile traffic offloading scenario consists of six major steps: offloading initiation, context collection, offloading decision, network association, data transmission, and offloading termination.

1. *Offloading Initiation* – The offloading procedure can be initiated by the network side (network-driven offloading), or by the mobile system

[15] Wi-Fi Direct, WiFi Alliance: `http://www.wi-fi.org/discover-and-learn/wi-fi-direct`

(user-driven offloading). Network-driven offloading can be triggered by dedicated signaling protocols such as router advertisement [76], enabling operators to dynamically manage and balance the traffic load. User-driven offloading is often triggered by applications that need to access the Internet for content, which is based on the demand of the user.

The network-driven offloading introduces overhead in terms of extra signaling and potential energy cost, but it can offer timely and optimized offloading guidance based on the comprehensive knowledge from the network side, such as network structure and condition. On the other hand, user-driven offloading avoids the extra signaling cost but lacks network context, making it less efficient for users at high moving speed.

In the initiation phase, it is important to keep the existing data flows uninterrupted to guarantee consistent service experience. For data flows that are sensitive to connectivity interruption, one good practice is to keep using the existing channel until the ongoing flows complete and then offload new flows to the new channel. For flows that can tolerate connectivity interruption, we could apply the delay-transfer scheme [5] to postpone the data transmission for a delay tolerance threshold and then offload the ongoing flows to the new channel once the connectivity is established.

2. *Context Collection* – The context information is essential for mobile traffic offloading as input to make the offloading decision. Users can obtain context information either from network operators or from their surrounding access environment. The key information includes the user location, potential offloading targets, condition of access network and connection details such as extended service set identification (ESSID) or MAC addresses, signal-to-noise ratio of WiFi access points, and wireless fingerprint information [77].

   The collected context information will be sent to either remote controlling servers or local management components. For the remote option that utilizes cloud support, a dedicated signaling channel is required such as cellular data connection. Therefore the proposals relying on remote support are limited by the channel condition, especially when such a channel is congested. This also affects the scalability due to the dependence on a centralized entity. Compared to the remote approach, the local solution does not depend on external entities. However, by relying solely on local resources, context information can be incomplete or less accurate compared to the remote option.

3. *Offloading Decision* – The decision process involves computation according to the pre-defined algorithms and operation logic, and delivering control messages to mobile devices to carry out traffic offloading. By taking the context information as input, an offloading decision can be made either at the network side or using local resources on the mobile device.

   By offloading the computation to the network side, we can improve efficiency in terms of latency by using the powerful hardware. However, this approach depends on the infrastructure support and requires network connectivity. On the other hand, local decision is more flexible and robust to network conditions, but at the cost of local resources such as energy. The local operation also suffers from the limitation that there is limited external knowledge available for improving the accuracy of offloading decisions.

4. *Network Association* – Based on the offloading decision, mobile devices need to perform network association to enable traffic offloading. The association process includes access/peer discovery via pre-defined configuration protocols such as DHCP [78] and DNS [79, 80] to establish connectivity to the target offloading networks.

   When users are moving at high speed, the connectivity period for offloading is often short. This demands an efficient association supported by optimized protocols to avoid excessive association cost, which can decrease the time for data transmission.

5. *Data Transmission* – As the key part of mobile traffic offloading, data transmission determines how much data can be offloaded from the congested mobile networks in order to improve the overall service quality. Depending on the type of traffic such as real-time streaming, delay-tolerant traffic and web surfing, the offloading design can utilize the characteristic of traffic for optimization.

   In the short period of offloading, which is typical for mobile users, the bitrate and condition of the wireless access can affect the offloading efficiency. At the same time, the offloading design also needs to consider the hardware limitation on existing mobile devices such as restricted size and output of wireless antenna.

6. *Offloading Termination* – A successful offloading session should be terminated by closing the temporary offloading connection and smoothly

switching to other available networks. The prior research on handover mechanisms has illustrated how to seamlessly migrate from one access network to another [55], [63]–[68], [81]–[84].

To achieve efficient and smooth termination, guidance can be obtained from the network side or from the local heuristic prediction to spot potential connectivity [85, 86, 87]. The termination process is one of the key factors affecting the adoption of mobile traffic offloading.

### 2.2.2 Solutions review

We provide an overview of research proposals and industrial solutions for mobile traffic offloading.

***Research Proposals and Investigations***

The pioneering research on mobile traffic offloading focuses on network measurement and experiments to improve our understanding of the feasibility, impact, and incentives of traffic offloading.

To explore the feasibility of WiFi-based offloading, researchers recruited 100 iPhone users for two weeks from metropolitan areas of South Korea to track WiFi connectivity in February 2010[6]. The simulation study based on the collected traces revealed that WiFi is able to offload 65% of the total mobile data traffic for average users without using any delayed transmission. The offloading efficiency can be further improved if the transmission can be delayed until a user enters a WiFi-covered area. By showing the upper bound of performance gain, this work presented the value and effectiveness of WiFi offloading for 3G mobile networks.

To further understand how effective traffic offloading can perform in the wild, researchers in Japan have conducted a two-day measurement over 400 Andriod smartphone users [89]. This study provided several findings based on the collected dataset. First, the total amount of traffic volume via WiFi was much larger than that of 3G, indicating the effectiveness of WiFi offloading. Second, a large fraction of traffic offloaded to WiFi came from a small number of users. For instance, the top 30% of data users downloaded over 90% of their total traffic volume over WiFi. Third, comparing with the traffic volume offloaded via home WiFi, the volume offloaded to public WiFi was relatively low. Moreover, offloading traffic through WiFi was common during the weekend and in the evenings of weekdays. The volume of traffic offloaded to WiFi was low during the weekday rush hours. There were 20% of users who only used 3G networks and over 50% of users turned off their WiFi interfaces during business hours. This measurement study confirmed the feasibility of WiFi offloading and indicated that there is room to improve the current situation by promoting users to utilize WiFi more effectively.

To augment 3G capacity with WiFi, researchers developed a system called Wiffler [5] which can leverage the delay tolerance of applications to improve access performance by using WiFi networks. The design of Wiffler was based on the extensive measurement of 3G and WiFi on moving vehicles in three different cities. The measurement results showed that the average 3G and WiFi availability across those cities was 87% and 11%, respectively. To overcome the poor availability of WiFi, Wiffler adopted a fast switching mechanism to allow the system to quickly switch to 3G when the performance of WiFi can not satisfy the application requirements. According to the results from the trace-driven simulations, Wiffer can reduce 3G usage by 45% for a delay tolerance of 1 minute.

Shevade et al. proposed a system called VCD [7] to enable high-bandwidth content distribution over WiFi for the vehicular networks. VCD can save 3G usage by replicating content to candidate WiFi access points so that users can fetch the content via those WiFi APs instead of using 3G. A dedicated mobility prediction algorithm was proposed to enable the proactive content pushing to improve the efficiency of WiFi usage. In a similar manner, Ristanovic et al. also designed an algorithm named HotZones [11] that exploited the delay tolerance and user mobility to enable efficient WiFi offloading.

Hou et al. [8] proposed a transport layer solution named oSCTP to support WiFi offloading for vehicular networks. As an extension of SCTP [90], oSCTP was capable of using multiple network interfaces for data transmission. By using a utility and cost-based formulation to decide the suitable offloading volume, oSCTP used 3G connection as a backup when WiFi was of poor performance or unavailable. The results of driving experiments showed that oSCTP can achieve roughly 65%-80% overall load reduction on 3G by exploiting a metro-scale WiFi network.

Regarding the incentive of mobile traffic offloading, Zhuo et al. proposed an incentive framework, named Win-Coupon, to motivate users to offload their mobile traffic to other networks such as WiFi [9]. To minimize the incentive cost, two important factors were highlighted: the delay tolerance and the offloading potential of the users. The proposal exploited the trade-off between the amount of offloaded traffic and the users' satisfaction affected by the delay. It utilized a reverse auction mechanism to dynamically determine the offloading solution based on the delay tolerance and the offloading potential of users. Based on extensive trace-driven simulations, the results revealed the impact of various factors on traffic offloading, such as the number of users in the system and the delay tolerance level of users.

Besides WiFi-based offloading schemes, utilizing opportunistic communications was also investigated. Han et al. [91, 92, 93] first explored the potential usage of opportunistic communications for mobile traffic offloading through a case study on the target-set selection problem. By exploring the social participation in mobile social networks, three algorithms were proposed to identify critical users in content delivery. The results of trace-driven simulations showed that opportunistic communications can help offload mobile traffic by up to 73.66%. Several research proposals [10, 11, 94, 95, 96] also adopted the opportunistic offloading to their design and combined it with social relationship and mobility prediction. To understand the incentives of opportunistic-based solutions, Cambridge researchers explored the coordinated resource sharing among co-located users and analyzed the benefits and challenges [97]. A large-scale user study [98] demonstrated the potential for opportunistic communications with respect to mobile devices in practice.

Regarding connectivity and deployment, measurement studies on WiFi access showed that grassroots WiFi networks were viable for applications that can tolerate intermittent connectivity [99, 100, 101, 103]. An extensive head-to-head performance comparison of 3G and WiFi in New York revealed the throughput and coverage characteristics of different access networks [102]. To achieve efficient traffic offloading, Bulut et al. studied the problem of WiFi access point deployment and proposed a deployment algorithm based on the density of data request frequency [12]. To exploit the diversity of multi-access environments, MAR [104] and Kibbutz [105] provided system-level solutions to leverage available network interfaces and connections.

The recent work on code and computation offloading [106] is an important topic for mobile networking. The key proposals include MAUI [107], Cuckoo [108], CloneCloud [109], and ThinkAir [110]. As the focus of this dissertation is on traffic offloading, the computation offloading is outside the scope of our discussion.

#### ***Solutions from Standardization Bodies***

To enhance network and traffic management, 3GPP have worked on several offloading solutions for their network architecture starting from the 3GPP Release-9 [16]:

[16] 3GPP Rel. 9: `http://www.3gpp.org/specifications/releases/71-release-9`

- Multiple Access PDN Connection (MAPCON) [111] allows for a UE to connect to different packet data networks (PDN) simultaneously via a 3GPP access and a non-3GPP access. MAPCON essentially enables traffic offloading from the mobile access to alternative accesses such as WiFi. In MAPCON, the traffic is routed through the operator's core network and GGSN/P-GW. The routing rules and policies [112] for traffic offloading can be conveyed via ANDSF [58].

- Selective IP Traffic Offload (SIPTO) [113] allows for the core network to select a GGSN/P-GW topologically or geographically close to a UE to offer more optimized routing of IP traffic. When the UE moves too far away from the associated GGSN/PGW the network can force re-establishment of the PDN connection and select a more optimal GGSN/PGW.

- Local IP Access (LIPA) [113] allows for the use of Local P-GW (LGW) located in a home evolved NodeB (HeNodeB). In LIPA, the IP addressing scheme used by the LGW is local and managed by the LGW/H(e)NodeB rather than the home operator. The change of the LGW will result in re-establishment of the PDN connection and a disconnection in the IP connectivity. The concept of LIPA is similar to SIPTO in that it forces UEs to select more optimal LGWs and hence avoid overloading the congested LGWs.

- IP Flow Mobility (IFOM) [114] extends the Dual-Stack Mobile IPv6 (DSMIPv6) [115] to flow-based mobility. In IFOM, the routing rules and policies are managed by ANDSF, and IFOM offers IP address preservation while switching the point of attachment to the network. IFOM is a superset of MAPCON in which an IP flow can be moved selectively to any available access.

- S2a Mobility based on GTP & WLAN access to EPC (SaMOG) [116] allows for offloading traffic from the mobile access to WiFi access by integrating the managed WLAN access into the EPC. In SaMOG, the traffic still gets routed through the operator's core network and GGSN/PGW. The integration requires 3GPP-specific functions in the WLAN access network and core network gateways. When UEs switch between WLAN and 3GPP accesses, SaMOG does not offer IP address preservation.

- S2b solution [58] allows for accessing the EPC over an IPsec tunnel from non-3GPP networks. It can essentially enable traffic offloading from the cellular access to WLAN, while the traffic still gets routed through the operator's core network and GGSN/PGW.

Besides the solutions that aim at seamless mobility, non-seamless traffic offloading is the simplest form of 3GPP solutions [26, 117]. In the non-seamless traffic offloading, the offloading related operator policies for IP interface selection (OPIIS) [112] can be managed by the ANDSF, by the OMA Device Management (DM), or by a local configuration. The traffic can be directly routed to a WiFi access to bypass the operator packet core and mobile access. The non-seamless solution reflects what a multi-interfaced UE can do today. A detailed review of the existing traffic offloading schemes is available in the survey study [118].

### 2.2.3 Technical and business perspectives

There are two technical motivations that drive the development of mobile traffic offloading: 1) To complement the coverage and capacity of mobile access with a cheaper technology but still route the traffic through the operator's packet core. 2) To bypass the operator's packet core, mobile access, and possibly the backhaul completely in order to maximize the offloading volume. 3GPP has provided solutions to address both cases. For example, MAPCON can be used for traffic offloading but the offloaded traffic is still routed through the operator's core network. Meanwhile in SIPTO, it is possible to bypass the core network and thereby allow the traffic to flow directly from the non-3GPP access to the Internet. On the other hand, the existing research proposals mainly target the latter one by using public wireless networks such as WiFi.

An important topic for mobile traffic offloading is how to evaluate its effects and impact. The question can be illustrated from two angles: the operator perspective and the user perspective. The operator perspective mainly concerns the technical and business values of incumbent mobile network operators. It covers Quality of Service (QoS), resource utilization, capital expenditure (CAPEX), and operational expenditure (OPEX). A key metric for the operator perspective is offloading efficiency which is defined as the ratio of bytes offloaded to other channels, such as WiFi or femtocells against the total bytes generated by the mobile systems [6]. The user perspective concerns the values of end users, covering user service experience and energy consumption. Key metrics for the user perspective include the energy consumption on devices and performance indicators that depict the Quality of Experience (QoE) such as throughput and latency.

Considering access technologies, both WiFi and femtocell are promising candidates to support mobile traffic offloading, while each technology presents its relative advantages and disadvantages. For WiFi-based design, the fast evolving 802.11 standards and the vast WiFi deployment in public areas and home environments makes it appealing in terms of technical maturity and deployment cost. As WiFi operates in unlicensed bands, it allows for operators to access much a larger spectrum and to have flexibility in solution deployment. However, since public WiFi networks may not be managed by the mobile operators, it can be difficult to coordinate and control traffic offloading. WiFi offloading may increase the battery drainage on mobile devices [92]. As an extension of the macrocells of mobile networks, femtocells can achieve reduced transmit power while maintaining good indoor coverage. The femtocell based offloading does not require extra network interfaces comparing to the WiFi-based offloading. Because mobile operators typically deploy and manage the femtocell networks, there are fewer obstacles for femtocell-based offloading schemes to be integrated to the mobile networks. However, careful planning is still needed as femtocells operate in the licensed bands, which may cause interference. More detailed discussion on femtocells is available in the survey study [119].

Regarding the business perspective, mobile traffic offloading has direct impact on the CAPEX of mobile operators with potential monetization opportunities. First, by offloading traffic to other access networks, the congestion on the mobile access can be relieved. This allows for mobile operators to save the expenditure of infrastructure upgrade in order to accommodate the surge of mobile traffic. Second, offloading solutions such as WiFi offloading provide a low-price alternative for mobile operators to expand access capacity in dense areas with high capacity demand. Rather than investing the more expensive cellular infrastructure, mobile operators can extend their access networks by deploying WiFi hotspots, which is becoming popular nowadays. This approach also helps cut the OPEX needed otherwise for maintaining the cellular infrastructure. Third, traffic offloading can create incremental revenue opportunities for mobile operators either through value-added services or by combining with the mobile data bundles. Moreover, it can help create business opportunities and new markets for startup companies [17].

## 2.3 Mobile Energy Awareness

In this section we present an overview of mobile energy awareness and discuss how to achieve energy-aware traffic offloading.

[17] Wi-Fi Offloading Solution, NOTAVA: `http://notava.com/en/home`

### 2.3.1 Techniques for mobile energy awareness

The main goal of mobile energy awareness is to improve energy efficiency in mobile systems by striking a balance between energy saving and service experience. Compared with traditional services such as voice call and short message service, modern mobile applications consume more computing and networking resources and therefore demand much more energy. Because battery technology has developed slowly compared to the development of hardware and services on mobile systems, battery life has become a critical bottleneck for users to fully benefit from the fast evolving mobile computing technologies. This challenge has driven the research development for techniques to improve mobile energy awareness.

We present the related work in this domain, including *measurement studies and modeling*, and *optimization techniques and tools*. We focus on solutions that have an impact on mobile traffic offloading.

#### *Measurement and modeling*

To develop energy-aware techniques, the first step is to understand how energy is consumed in mobile systems. Two practical approaches for achieving this goal are power measurement and power modeling. Power measurement focuses on providing an empirical view on how much energy is consumed, while power modeling focuses on using mathematical models to describe how the energy is consumed. These two approaches complement each other.

There is a rich body of research investigating energy consumption in mobile systems through measurement studies, ranging from hardware components, operating systems, to mobile applications. For instance, Carroll and Heiser conducted a measurement study to understand where and how the energy was used on a modern mobile device [120]. Their results provided a breakdown of energy consumption for hardware components such as CPU, memory, touchscreen, and network interfaces. Rice and Hay also examined the energy consumption of two Android handsets and proposed a power measurement system that can produce annotated traces of system and network activities [121]. Trestian et al. conducted a measurement study in the testbed to explore the energy consumption of video-centric applications [122]. Chandra has analyzed the network behavior and energy consumption characteristics for popular multimedia streaming formats [123]. Several studies also looked into bandwidth predictability, traffic patterns and usage diversity in a mobile environment to understand their impact on energy consumption [124]–[129].

Because cellular and WiFi are the dominant communication technologies used by mobile systems nowadays, researchers have conducted extensive measurements to understand their characteristics and impact on energy consumption [130]–[145]. Balasubramanian et al. conducted an extensive measurement study on the energy consumption characteristics of 3G, GSM, and WiFi [130]. Their results revealed that 3G and GSM incurred a high tail energy overhead due to the high power states after data transmission. The association overhead of WiFi was comparable to the 3G tail energy overhead, but WiFi was much more energy efficient than 3G in data transfer.

Several studies have explored the energy consumption characteristics in cellular networks [131]–[139]. For example, Auçinas et al. investigated the energy consumption of mobile applications that provide continuous online presence [131]. Their measurement results revealed that there was an excessive cost for mobile applications to maintain online presence in terms of signaling and battery life. Even without user interface activity, maintaining online presence can increase the power drain by up to 9 times compared with no online presence. The power characteristics, performance, and radio resource allocation in 2G, 3G and 4G/LTE have been studied in depth through several projects [132, 133, 134, 135, 136]. At the same time, there were measurement studies that investigated the traffic characteristics of both data and control planes in cellular networks [137, 138, 139]. The experience and observations from those measurements provide guidelines for network operators and developers to improve energy awareness in network management and mobile applications.

Besides cellular technology, researchers have studied WiFi and its energy consumption in depth [140, 141, 142, 143, 144, 145]. For instance, Halperin et al. first reported their empirical characterization of the energy consumption of 802.11n across a broad range of configurations [140]. Their results showed that for shorter packets it was more energy efficient to use the single antenna operation. Only for large packets and when the channel was good enough to support the highest multiple-input multiple-output (MIMO) rates, the use of multiple radio frequency chains can be energy efficient. An experimental investigation by Garcia-Saavedra et al. provided a detailed anatomy of the per-packet energy consumption of 802.11 [143]. Their analysis revealed that a substantial fraction of energy was consumed when packets were processed through the protocol stack and such energy cost should be considered in the power modeling. The recent study on 802.11ac presented a detailed characterization for energy consumption and interference in both indoor and outdoor environments [145]. Their study identified new throughput and fairness anomalies that were introduced by using a larger channel width. The unplanned selection of primary channels and channel widths can severely degrade the throughput of links that operated at larger channel widths. Their results also showed that increasing the number of spatial streams can be more energy efficient compared with increasing the channel width in achieving the same percentage increase in throughput.

Researchers have investigated how the security mechanisms affected the energy consumption in the past [146, 147, 148, 149]. Potlapally et al. provided a comprehensive analysis of the energy consumption of cryptographic algorithms that form the building blocks for security protocols [146]. They observed that the energy consumption of asymmetric algorithms depended on the key size, but that of symmetric algorithms was not significantly affected by the key size. Another insight was that the level of security provided by a cryptographic algorithm can be traded off for energy savings by tuning parameters such as the key size. The work by Miranda et al. investigated the energy consumption of Transport Layer Security (TLS) [149]. They observed that the energy consumed during the handshake phase was dependent on the the length of the server certificate and transmission round trip time (RTT). For transactions less than 10 KB, the TLS overhead can account for over 60% of the total energy consumption.

Energy modeling and profiling have been researched extensively to improve our understanding of how energy consumption is related to hardware activities and software operations [150]–[157]. Flinn and Satyanarayanan proposed a profiling tool PowerScope that can map energy consumption to program structure and provide fine-grained profiles of energy usage of process and procedure [150]. Another pioneering work, Joule Watcher, can offer fine-grained profiling at thread level by using counters embedded in the hardware drivers to register events that imply energy consumption [151]. Li and John have proposed a set of models to estimate the run-time energy dissipation at operating system level [152]. By using the built-in battery voltage sensors commonly deployed on smartphones, Zhang et al. proposed a power model generation tool PowerBooter and an on-line power profiler PowerTutor that can determine component-level power consumption during application excution [153]. Pathak et al. proposed a power modeling approach based on tracing the application system calls [154]. Their system-call-based power model can capture both utilization-based and non-utilization-based power behaviors of I/O components and thereby improve the accuracy of energy estimation. The proposals of Sesame [155], DevScope [156] and V-Edge [157] represent a recent trend to achieve efficient self-constructive power modeling without any external assistance. A comprehensive examination of power modeling is available in the book "*Smartphone Energy Consumption: Modelling and Optimization*" [19].

### *Optimization techniques and tools*

Motivated by the results from measurement and modeling, several techniques and tools have been proposed to optimize energy usage and power management on mobile systems [158]. We highlight the key proposals below with emphasis on solutions relevant to mobile traffic offloading.

Regarding energy-aware operating systems, Cinder [159] is a mobile OS based on HiStar [160] that aims to provide efficient energy allocation to applications. Cinder adopts two abstractions named Reserves and Taps to enable controlling and accounting for energy. In Cinder, a *reserve* describes the right to use a given amount of energy. When an application consumes the energy, the Cinder kernel will reduce the values in its corresponding *reserve*. The rate at which applications can consume the *reserve* is controlled by the *taps*. The role of a *tap* is to transfer the energy between *reserves*. Once an application has consumed all its *reserves*. the Cinder kernel can prevent it from performing more actions. By adopting a different approach, ErdOS [161] is a user-centered mobile operating system that aims to improve energy awareness by managing resources proactively and exploiting opportunistic access to resources in nearby devices using the social connections. ErdOS can monitor the resource state, application demand, and user interaction pattern. By learning the user activities, ErdOS predicts the future demand and manages the available resources in an event-based manner. There are also proposals like EcoSystem, Odyssey, and CondOS that aim to enable energy awareness at the operating system level by using energy-aware scheduling [162], adjusting application behavior to energy demand [163], or managing sensor resources [164].

Besides operating system level solutions, there is a rich body of research investigating application [165, 166, 167, 168], middleware [169]–[175], transport protocol [176, 177], network interface [179]–[185], and cross-layer optimizations [186, 187, 188, 189]. To highlight, AppScope [167] is an application energy metering framework based on DevScope [156] that can provide accurate process-level energy information. To achieve efficient metering, AppScope uses kernel activity monitoring to trace system calls and applies a linear model-based estimation to derive the energy consumption. The middleware solution STPM [170] supports energy-aware resource management by leveraging collaboration between applications and operating system. In STPM, applications disclose indications, named ghost hints, about their intentions to use I/O devices. The underlying system exposes context information about its general state. By collecting information from applications and system, STPM can efficiently adjust the power management strategy according to the observed pattern. At the transport level, Pluntke et al. [177] proposed to use Multipath TCP [178] to improve the energy efficiency in transport performance. They designed a method for generating multipath schedulers by formalizing them as Markov decision processes. The inputs for this method include the energy models for network interfaces and application models obtained from monitoring users' communication sessions. The generated schedulers will take over the Multipath TCP optimization from the current scheduler instance. Cell2Notify [185] is an energy management architecture that uses the cellular connection on a mobile system to wake-up its WiFi interface upon an incoming VoIP to redirect the call through the WiFi interface. Cell2Notify leverages the energy efficiency of WiFi for data transmissions while minimizing its expensive scanning cost. To reveal hidden energy bottlenecks in the resource constrained mobile execution environment, Qian et al. proposed Application Resource Optimizer (ARO), a tool that can accurately expose cross-layer interactions [188] ranging from higher layers such as user input down to the lower layers such as radio resource control. The evaluation on popular mobile applications showed that ARO can discover previously unknown insufficient resource usage, which was mainly due to a lack of transparency in the lower layer protocol behavior. With the captured cross-layer information, especially the characteristics of cellular connection, ARO can also help developers to build applications that are cellular-friendly and energy efficient.

For cellular access, researchers have proposed several mechanisms to reduce energy consumption on mobile systems, which can be grouped into two categories: scheduling optimization [130, 190, 191, 192] and operation mode adaptation [193, 194, 195, 196]. The approaches for scheduling optimization aim to improve energy efficiency by using dedicated data transfer schemes such as prefetching and delayed transfer. On the other hand, the approaches for operation mode adaptation mainly manipulate the Radio Resource Control (RRC) state transition of cellular interface to strike a balance between latency and power saving. A well-known mechanism in scheduling optimization is TailEnder [130]. TailEnder leverages the delay tolerance of applications and utilizes techniques such as batching and prefetching to minimize the cumulative energy consumption while meeting the user-specified deadlines. TailTheft [190] is a traffic queuing and scheduling mechanism that shares the same design principle as TailEnder. By using the proposed virtual queuing and delayed transfer, TailTheft can batch traffic among different applications and share the tail-time energy among them. Similar to TailEnder, TailTheft requires applications to specify how much delay is acceptable. Proposals like Bartendr and LoadSense utilize prediction-based scheduling driven by signal strength [191] and base station power level [192] to carefully coordinate and guide data transfers. Both Bartendr and LoadSense have shown the potential to improve energy saving by up to 60%. In the operation mode adaption category, Deng and Balakrishnan developed a mechanism that can adapt radio state transitions according to traffic load by learning the patterns and predicting when a traffic burst would start or end [193]. To overcome the signaling overhead generated by frequent fast dormancy operations, RadioJockey [194] utilizes the application execution traces to predict the end of communication sessions and thereby accurately invokes fast dormancy. Researchers also analyzed the usage of Discontinuous Reception (DRX) in LTE and LTE-Advanced networks and proposed the DRX switching scheme to strike a balance between latency and energy saving [195, 196].

Regarding the energy-aware optimization for WiFi, the problem setup is different from cellular access because of different tradeoffs we aim to balance. In WiFi, the time and energy consumed by state transition is negligible compared to those for cellular access. Therefore, an important topic for WiFi power management is to dynamically determine the sleep duration when the WiFi interface is off. Micro power management ($\mu$PM) [197], SOFA [198] and SleepWell [199] are optimization schemes that regulate the WiFi sleeping circles to improve energy saving. The proposal $\mu$PM improves energy-saving by putting the network interface into power saving mode during idle intervals as short as several microseconds [197]. As a client-centric approach, $\mu$PM determines when to wake up the interface and how long to keep it awake to receive retransmitted data. $\mu$PM makes decisions based on the history-based prediction of the next transmission and the load of the access network. SOFA considers the scheduling policy on the access point (AP) side and aims to save energy on the client side by minimizing the time clients are forced to stay awake while downlink traffic is being transmitted to other clients [198]. In a similar manner, SleepWell is an AP-centric mechanism that aims to avoid network contention [199]. In SleepWell, APs regulate the sleeping window of the associated clients in such a manner that different APs are active during non-overlapping time slots. The wisdom behind SleepWell is the common rule-of-thumb to avoid rush hours that we go late to the office and come back late, as well. By reducing the rush, we can open up more free time for all, and yet keep the total working hours unaffected. To reach a balance between throughput and energy saving, rate adaptation and network assistance approaches have also been explored in the past, by leveraging energy modeling [200], delayed transfer [201], antenna configuration [202], and network assistance such as AP virtualization [203] and proxying [204].

Besides system and networking optimization, diagnosing and monitoring tools are valuable in revealing abnormal battery drainage and providing viable suggestions to regular mobile users and application developers. For example, Eprof [205] can detect energy bugs in mobile applications through its fine-grained per-routing energy profiling, which exposes abnormal battery usage at the system call level. In Eprof, a new accounting presentation for I/O energy consumption, named bundles, is introduced to assist developers to quickly understand and optimize the energy drain of their applications. Given the abnormal battery drain problem caused by software defects or misconfiguration, eDoctor [206] provides a diagnosing tool to help regular users troubleshoot the battery issues on their mobile devices. In eDoctor, the execution of an application is divided into execution intervals, which are grouped into phases. The intervals in the same phase share similar resource usage patterns. When an application starts consuming energy abnormally, its execution results in new phases that do not appear in the normal execution. By combining such phase execution information and configuration changes, eDoctor can accurately identify the issues and further suggest a set of repair solutions to end users. For cellular access, Rilanalyzer [207] is a monitoring tool to facilitate researchers to analyze cellular access issues without requiring access to internal components in the cellular networks. By recording the low-level cellular radio state, cellular network control-plane data, and user-plane data, Rilanalyzer is capable of identifying overlooked issues and reveal how different network configurations interact with applications leading to network and energy overhead. Carat [208] is a collaborative mobile energy awareness system that can detect and diagnose energy anomalies on modern mobile systems. Based on its large datasets with over 100 million samples from over 725,000 devices, Carat uses a collaborative inference method to derive regularity and deviation of application energy usage from the community and leverages such insight to characterize abnormal energy usage. A detailed discussion on collaborative mobile energy awareness is available in [209].

### 2.3.2 Energy awareness in mobile traffic offloading

The impact of mobile traffic offloading can be evaluated from two aspects: the operator perspective and the user perspective. The ultimate goal is to benefit both network operators and users. However in reality, the main concern of network operators is about how to offload as much traffic as possible to alleviate the pressure on their network infrastructure, whereas a key factor from the user perspective is the battery life. Although much of the prior work in traffic offloading focused on the operator perspective, recent studies have looked into the user perspective and shown that by improving energy awareness in traffic offloading we can extend the battery life for mobile users and still provide a desirable outcome from the operator perspective [15, 16].

To get a better understanding of energy cost in mobile traffic offloading, we analyze the offloading procedure and further suggest viable approaches as highlighted in Table 2.1 to enhance energy awareness.

- In the initiation phase, if offloading is triggered by the network side, signaling consumes energy. If signaling messages are delivered too frequently with a large volume of data, such unintentionally recurrent interactions may promote the cellular interface state, thereby leading to excessive draining of the battery [131]. This problem is comparable to the case of background traffic. On the other hand, the user-driven initiation can not benefit from the proactive guidance but it does not consume extra energy in this phase.

  Based on the observation, it is crucial to avoid frequent network signaling with large payload, which can lead to extra energy consumption due to the change of cellular radio state. In order to benefit from the network support and to support dynamic traffic management, an adaptive scheme that combines the user-driven and network-driven initiation is recommended. Such a scheme helps strike a balance between the efficiency and energy consumption [16, 92].

- The context collection phase has to be done carefully. As revealed in a previous study, frequent scanning of available WiFi networks consumes a significant amount of energy [92]. If GPS is used, a cold-start can take around 20 seconds and consume ~6.3 Joules [16]. If the context information is delivered to a remote server, it also costs energy.

  In the context collection phase, constant scanning must be avoided if possible for its high energy consumption on mobile devices. Due to the relatively high energy consumption and the long latency to obtain a fix from the cold-start, the GPS usage in traffic offloading needs to be assisted by energy efficient design, such as using techniques proposed in [210, 211]. To support energy efficient positioning, WiFi-based positioning can also be used as an alternative. For context processing and management, it is recommended to share the load between local devices and remote servers, which can fully utilize the knowledge of access network and infrastructure, as well as the computing resources of both sides.

- In the decision making phase, if the computation is done locally,

Table 2.1: Approaches to enable energy-aware traffic offloading.

| Phase | Recommendations |
|---|---|
| Offloading Initiation | 1) Avoid frequent signaling from the network side that triggers cellular state change.<br>2) Combine the user-driven and network-driven in an adaptive manner to improve efficiency. |
| Context Collection | 1) Avoid unnecessary scanning and GPS operation.<br>2) Utilize an energy efficient positioning mechanism such as WiFi fingerprinting.<br>3) Adapt the context management for both local and remote processing to strike a balance. |
| Offloading Decision | 1) Utilize energy-aware algorithms to guide the decision.<br>2) Adopt dynamic mechanisms to update the logic according to the network condition.<br>3) Utilize the cloud support to offload the energy cost from intensive computation. |
| Network Association | 1) Avoid time-consuming association operation or protocol.<br>2) Utilize guidance from the network side if possible to assist authentication and access connectivity. |
| Data Transmission | 1) Adopt optimization schemes for different types of traffic.<br>2) Avoid transmission over unstable or low throughput wireless links by predicting the user mobility and network condition. |
| Offloading Termination | 1) Utilize hints if possible from either network side or local controller for switching between connections.<br>2) Avoid frequent terminations that can shorten the data transmission time. |

the computational complexity of the algorithm determines the energy consumption. If the computation is offloaded to remote servers, the data communication due to the exchange of context and decision messages will consume energy.

To enable energy awareness, energy consumption must be a key factor in making traffic offloading decisions. In this phase, a dedicated energy-aware algorithm can be used together with other factors such as network performance and offloading capacity. Due to the fast change of network conditions, the energy-aware offloading decision needs to be adaptive to adjust its operation logic. To save energy consumed by computation, it is recommended to offload intensive computational loads such as mobility prediction to the network side.

- In the association phase, the connection establishment that involves various hardware and protocol operations can consume a non-negligible amount of energy. For instance, DHCP alone can consume 4.8 Joules [16]. Therefore, when the user is moving at a high speed, the frequent associations accelerate the battery drain.

  In order to minimize the time spent in the association phase, it is recommended to use light-weight configuration protocols and to avoid the ones requiring complex message exchange. To assist authentication, network support can be utilized to deliver configuration information.

- In the data transmission phase, the amount of available bandwidth and wireless condition greatly affect the energy consumption. By taking WiFi offloading as an example, when WiFi signal strength is poor, offloading cellular traffic to WiFi can cost more energy [16].

  For data transmission, the first recommendation is to adopt optimization techniques tailored for characteristics of different types of traffic and thus maximizing the transmission throughput during the offloading period. For instance, data batching and energy efficient scheduling of delay tolerant traffic can effectively amortize the penalty caused by the tail energy [130]. Several optimization techniques related to this have been proposed in the literature [5, 190, 191, 192]. The second recommendation is to avoid using unstable or low throughput wireless connections by estimating the network condition and user mobility [85, 86, 87, 124], [212]–[216].

- The termination phase should be handled in a timely manner. Otherwise more energy can be consumed during the periods of poor connectivity and low bandwidth.

Since the offloading termination affects the data transmission and service quality, we can improve energy saving by utilizing proactive approaches to plan the termination and apply efficient handover schemes to ensure the service continuity. There has been plenty of work on this [55], [63]–[68], [81]–[84]. To avoid frequent unnecessary terminations that degrade the transmission performance, it is recommended to utilize the knowledge from the network side (e.g., network setup and position of APs) combined with the mobility prediction techniques.

Mobile traffic offloading provides a promising way to alleviate the pressure on existing cellular networks overloaded by data traffic and it is gaining support in the future due to the rapid growth of mobile data traffic. By making the offloading process energy aware, it is possible to improve user experience by extending the battery life on mobile systems.

## 2.4 Software-Defined Networking for Mobile Networks

Software-defined networking (SDN) is a novel solution to network configuration and management. It articulates the vision of programmable networks by separating the control plane from the data plane with well defined programmable interfaces to provide a centralized global view of the network. The early commercial successes of SDN such as the wide-area traffic management system of Google [222] have attracted close attention from both industry and academia. Many of the leading information technology companies have joined SDN industry consortia such as the Open Networking Foundation and the OpenDaylight [18]. For its openness and programmability, SDN has gained significant traction in the mobile industry as well, being a promising direction to innovate the mobile network management. This section presents an overview of SDN and introduces the latest research development of SDN for mobile and wireless networks.

### 2.4.1 Overview of software-defined networking

SDN suggests separating the data and control plane with well defined programmable interfaces to simplify network configuration and management. In SDN, the control plane decides how to handle the traffic and the data plane forwards traffic according to decisions made by the control plane.

[18] OpenDaylight Project: `http://www.opendaylight.org/`

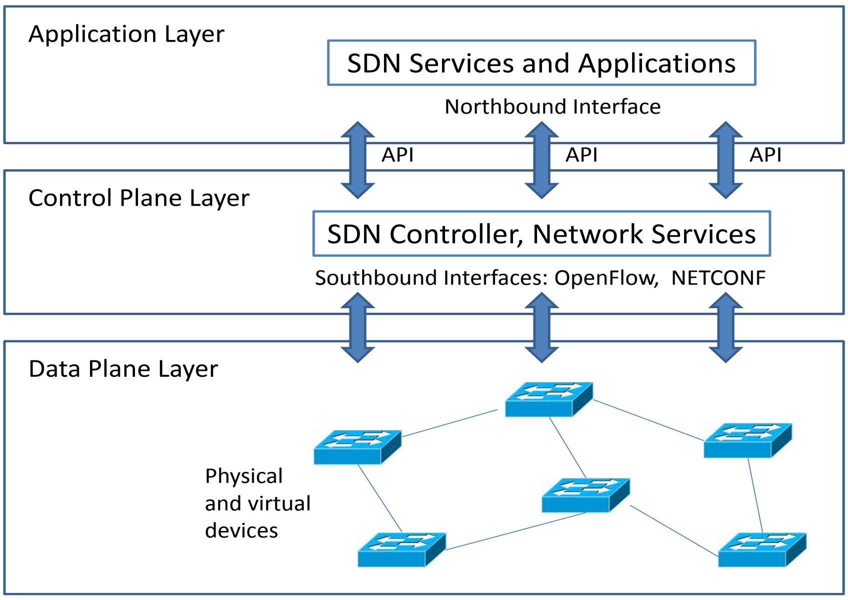

Application Layer
SDN Services and Applications
Northbound Interface
API
API
API
Control Plane Layer
SDN Controller, Network Services
Southbound Interfaces: OpenFlow, NETCONF
Data Plane Layer
Physical and virtual devices

Figure 2.5: SDN architecture.

### *SDN architecture and programmable interfaces*

As depicted in Figure 2.5 the architecture of SDN consists of three logical layers that are accessible through open APIs:

- *Data plane layer* consists of the physical and virtual devices that provide packet switching and forwarding as the connective fabric between all endpoints within the network.

- *Control plane layer* provides the consolidated control functionality for supervising the network forwarding behavior. Being the middle layer in the SDN architecture model, it implements multiple protocols for managing the physical hardware within the network and provides a set of common APIs to the application layer.

- *Application layer* consists of the business and network logic applications that control and monitor network behavior. In addition, more complex solutions can be implemented at this layer, such as applications for network resource orchestration and traffic management.

The programmable interfaces are essential to the layer design of SDN, allowing for abstracting the complexity of the underlying physical network and enabling the intelligence of centralized network management. In SDN, the southbound interface is capable of supporting multiple protocols such as OpenFlow [75], NETCONF [219], and SNMP [220]. It allows for controllers to expose the underlying device functionality to the upper layer services irrespective of protocols used between the controller and the network devices. The northbound interface is exposed by the controller to application developers to support their development of various SDN applications. A typical northbound interface offered by mainstream SDN platforms is the web-based REST API.

### *OpenFlow*

First proposed in 2008 [221], OpenFlow is a prominent example of SDN open interface that strikes a balance between the vision of fully programmable networks and the pragmatism to enable real-world deployment. The OpenFlow standard defines a protocol that allows the SDN controller to access an OpenFlow-enabled switch over the network. By relying on existing switch hardware, OpenFlow can be deployable immediately and is hence adopted quickly by the industry for accelerating the development of new SDN applications.

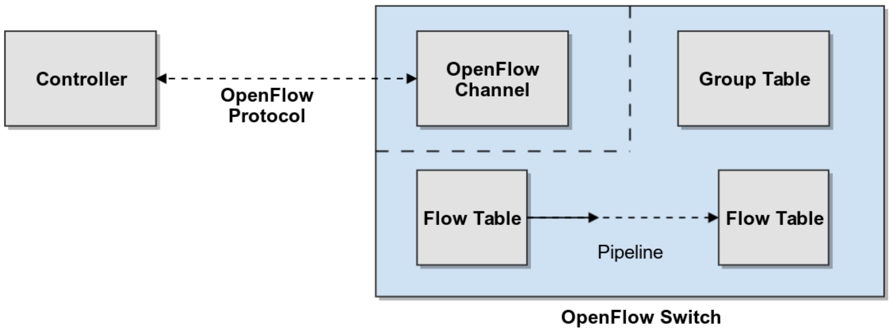


Figure 2.6: OpenFlow switch components.

The key components of an OpenFlow switch include the flow table, group table, and OpenFlow channel, as shown in Figure 2.6. The flow table and group table are designed for performing packet lookups and forwarding. The OpenFlow channel is the interface that connects each OpenFlow switch to a controller. Through this interface, the controller receives events from the switch and manages the switch. All the messages are formatted according to the OpenFlow protocol.

As specified in OpenFlow 1.4.0, each flow table contains a set of flow

Table 2.2: Flow entry in OpenFlow v1.4.0 [75].

| | | | | | |
|---|---|---|---|---|---|
| Match Fields | Priority | Counters | Instructions | Timeouts | Cookie |

entries. Each flow entry consists of match fields, priority, counters, instructions, timeouts, and cookies, as illustrated in Table 2.2. The match fields contain packet headers, ingress port, and optionally metadata (for passing information between tables). The priority field is for indicating the matching precedence of the flow entry. The counters field serves statistical purposes and is updated when packets are matched. The instructions field describes the OpenFlow processing that happens when a packet matches the flow entry. The timeouts field specifies the maximum time before the switch expires a flow and removes the entry from the flow table. The cookie field contains extra data values that can be used by controller to filter flow statistics, flow modification and deletion. A flow entry is identified by the match fields and priority and these two fields together can help a controller quickly identify a unique flow entry in the flow table.

An OpenFlow switch can contain one or multiple flow tables that are sequentially numbered starting at zero. The OpenFlow pipeline processing defines how packets interact with those flow tables and always start at the first flow table. Other flow tables may be used depending on the outcome of the first table. The group table in OpenFlow consists of group entries and each entry contains four fields: group identifier, group type, counters, and action buckets. The group entry contains a list of action buckets with dedicated semantics dependent on the group type. The group table enables OpenFlow to deliver additional methods of forwarding, such as to select a random port on a group of ports for load balancing. In general, packets are processed by switches supporting OpenFlow 1.4.0 in the following steps:

1. A switch performs matching and lookup for every incoming packet according to the flow tables, and if it is matched to one flow entry, the packet will be processed according to the actions defined in that flow entry.

2. If a packet could be matched to several flow entries, only the highest priority flow entry that matches the packet should be selected. The counters associated with the selected flow entry will be updated and the instruction set in the selected flow entry must be applied.

3. If no match is found, the switch will perform a default action such as dropping the packet or forwarding it to a remote controller for further handling.

The OpenFlow channel is the interface for connecting each switch to a controller. All OpenFlow channel messages must be formatted according to the OpenFlow protocol. The OpenFlow channel is usually encrypted using TLS [223], and optionally it can be run directly over TCP. By default, the OpenFlow channel is a single network connection. Alternatively, the OpenFlow channel may be composed of a main connection and multiple auxiliary connections to exploit parallelism. The auxiliary connections may use different transport mechanisms to the main connection, such as DTLS [224] or UDP. In OpenFlow 1.4.0, the OpenFlow protocol supports three message types: controller-to-switch, asynchronous, and symmetric. Controller-to-switch messages are initiated by the controller to directly manage or inspect the state of the switch. Asynchronous messages are initiated by the switch for updating the controller of network events and the switch state changes. Symmetric messages are initiated by either the switch or the controller.

#### *SDN control platforms*

Being a critical element in the SDN architecture, the SDN controller converts network-level policies/objectives into packet-level operations and provides an abstract view of the network to upper layer applications. Since the first OpenFlow controller NOX released in 2008 [225], a number of SDN control platforms have emerged, including POX [19], Ryu [20], Floodlight [21], and OpenDaylight [22].

As an example, Figure 2.7 presents the basic structure of a NOX managed network. This structure is shared by many other SDN control platforms, in which a set of OpenFlow switches are connected to and managed by a logically centralized SDN controller. NOX adopts an event driven design where an event represents the change in the system such as link failure or connection resume. The events can be generated by OpenFlow messages or low-level events. NOX uses the event handlers to capture and react to the events. Similar to the handlers for Unix system calls, developers can register handlers for particular events and trigger dedicated control logic upon the event occurrence. To manage the network in a simple and topology independent manner, NOX generates a centralized view for the managed network and constructs mappings of high level namespace to low level address. Besides using OpenFlow to manage underlying devices, NOX

[19] POX Platform: `http://www.noxrepo.org/pox/about-pox/`
[20] Ryu SDN Framework: `http://osrg.github.io/ryu/`
[21] Floodlight Controller: `http://www.projectfloodlight.org/floodlight/`
[22] OpenDaylight: `http://www.opendaylight.org/project/technical-overview`

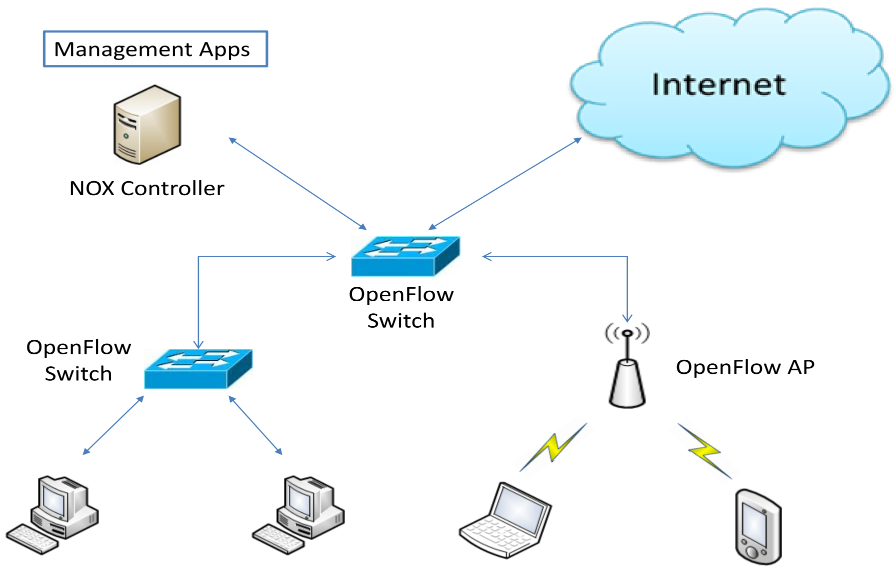


Figure 2.7: NOX architecture [225].

also provides a set of network libraries to support efficient implementation of common functions, such as DHCP and DNS.

In recent years, researchers have proposed novel controller design and techniques to improve the scalability of SDN systems. The notable proposals include HyperFlow, Kandoo, Onix, and ElastiCon:

- *HyperFlow* [226] is a distributed SDN control platform that allows more than one controllers to be deployed in the HyperFlow managed network. As shown in Figure 2.8, a HyperFlow network consists of a number of OpenFlow switches, several NOX servers running the HyperFlow control application, and an event propagation system for synchronizing the global view of the network. In HyperFlow, all the controllers share a consistent view of the network by using a publish/subscribe messaging paradigm. Each controller subscribes to three channels: the *data* channel, the *control* channel, and its own channel. All the controllers are allowed to publish to all channels. The HyperFlow application publishes selected events that are of general interest to the *data* channel. Commands targeted to a specific controller are published to the corresponding controller's channel. In addition, each controller periodically advertises itself to the *control* channel to facilitate controller discovery and failure detection. Hy-

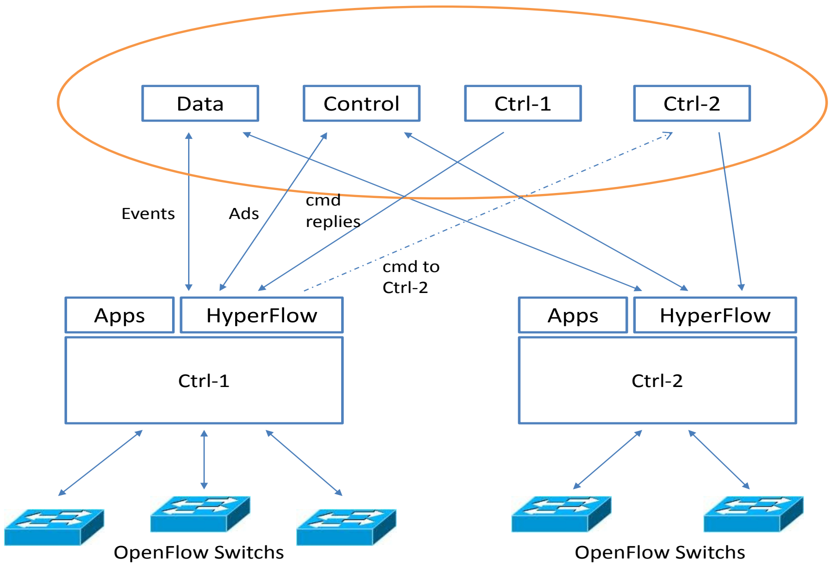


Figure 2.8: Overview of HyperFlow [226].

perFlow minimizes the control plane response time by keeping the network control logic centralized and localizing all decisions to each controller. The distributed design of HyperFlow enables the interconnection of independently managed OpenFlow networks and makes it resilient to network partitions and component failures.

- *Kandoo* [227] is a hierarchical SDN control framework based on OpenFlow. It introduces a two-level hierarchical architecture in which local configuration and management are separated from global policies. As shown in Figure 2.9, Kandoo utilizes two types of controllers to support its hierarchical design: the local controller and the root controller. The local controller directly manages a subset of switches and executes applications that do not require a global view of the network. The root controller is responsible for managing all the local controllers and implementing global policies. The root controller subscribes to events generated by local controllers. The local controller will then propagate those events to the root controller. By processing frequent events in local controllers and rare events in the root controller, Kandoo limits the overhead of frequent events on the control plane, thereby improving the scalability.

- *Onix* [228] is a distributed control platform that utilizes a distributed

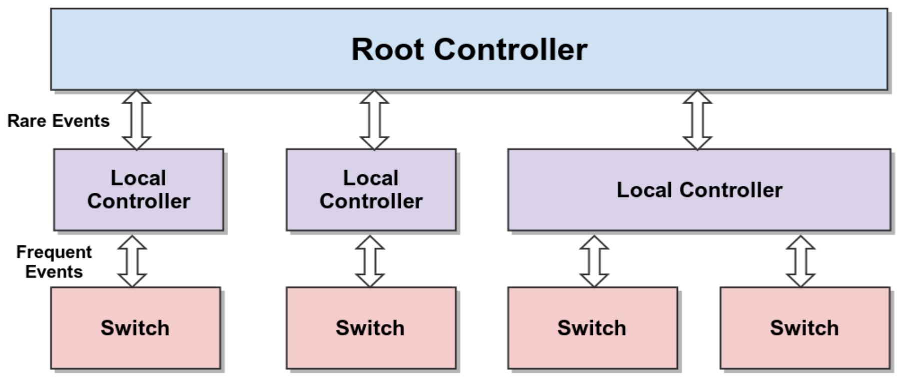


Figure 2.9: Kandoo framework [227].

data structure named Network Information Base (NIB) to represent network resources. In Onix, each network element is recorded as a set of key-value pairs with a globally unique identifier. Onix maintains a consistent NIB for the whole network through two mechanisms. First, Onix uses a transactional SQL-like database for persistent but less dynamic data. Second, Onix provides a one-hop, eventually-consistent and memory-only DHT system for the network state needing high update rates and availability. The key contribution of Onix is its general API for the control plane that allows for the development of scalable applications. By using the Onix platform, a network control plane can be implemented as a distributed system.

- *ElastiCon* [229] is an elastic distributed SDN control platform in which controllers can be added to or removed from the controller pool dynamically according to traffic conditions and the load is dynamically shifted across controllers. To support load adaptation, ElastiCon introduces a load estimation module that runs on the controller to report load statistics. ElastiCon sets the threshold for individual controller load and aggregated load of the controller pool. When the load on an individual controller is beyond the threshold but the aggregated load is within the aggregated threshold, ElastiCon will dynamically migrate selected switches to other idle controllers. When the aggregated load is above the aggregated threshold, ElastiCon can add new controllers to the pool.

Behind the SDN paradigm of separating the control plane from the data plane, we highlight four visible features and their benefits:

- *Centralized control and coordination* – The logically centralized control model is a key part of the SDN architecture, which mitigates the overhead from the traditional distributed protocol-based mechanisms. Although the centralized approach is often questioned for its scalability, it can deliver the changes of state and policy more efficiently in a managed domain compared with the distributed methods. The coordination feature also makes it possible that when one of the controllers fails, other standby controllers can take over the management tasks to avoid service breakage.

- *Programmability* – The programmability of SDN makes the implementation and deployment of new network functionality faster and easier, thereby speeds up the innovation at both hardware and software level. This agility can reduce the Operational Expenditure (OPEX) in that complex management tasks can be carried out by SDN applications in a coordinated manner, which helps save the maintenance cost. The programmability can also bring down the Capital Expenditure (CAPEX) by using software update instead of replacing the hardware to add new functionality, and hence avoiding the cost related to unnecessary hardware replacement.

- *Virtualized abstraction* – The layered design of SDN with well defined interfaces can hide the complexity of hardware devices from the SDN applications. Through the virtualized abstraction, a managed network can be divided into virtual networks that share the same infrastructure but are governed by different policy and security requirements. It helps promote the sharing, aggregation and management of available resources and enable dynamic reconfiguration and policy changes.

- *Openness* – The open standards promoted by SDN such as OpenFlow help build and develop open sourced communities that attract brain power and speed up the innovation. The openness combined with programmable APIs benefits networking research by enabling researchers to experiment their novel ideas through fast prototyping. It also benefits the interoperability and backward compatibility with legacy infrastructure and allows for different operators and providers to collaborate through the SDN framework.

Table 2.3: SDN solutions for mobile and wireless networks.

| | Cellular Environment | WLAN Environment |
|---|---|---|
| Edge Access | SoftRAN [232]<br>OpenRadio [233] | OpenAPI [236]<br>OpenRadio [233] |
| Core Network | CellSDN [230]<br>SoftCell [234] | OpenRoads [231]<br>Odin [235] |

### 2.4.2 Solutions for mobile and wireless networks

Given the phenomenal growth in mobile traffic and the inherent need to simultaneously operate over multiple wireless technologies, mobile network operators are seeking viable and cost-effective solutions to address these challenges. As illustrated in recent research [230, 231], one promising direction is to integrate SDN and fully utilize its features to refine the mobile network architecture. As highlighted in Table 2.3, we describe the latest SDN solutions for mobile and wireless networks that aim to address these challenges, covering both cellular and WLAN environments.

#### *Cellular Networks*

SoftRAN [232] and OpenRadio [233] are the latest proposals that apply SDN to innovate the mobile access networks. SoftRAN focuses on the control plane by abstracting multiple base stations into a virtual big base station. As shown in Figure 2.10, SoftRAN introduces a 3D resource grid to enable operators to manage the radio resources within a geographical area through three dimensions: time, frequency, and base station index. The SoftRAN controller maintains a RAN information base (RIB) which can be accessed by various control modules for radio resource management. The RIB consists of essential information to be updated by the controller, such as the interference map, flow records, and network operator preferences. Because the radio element has a more timely view of the local state than the remote controller, SoftRAN optimizes the control decision by splitting the control functionality between the central controller and the radio elements.

OpenRadio [233] proposes a data plane solution to enable programmability through its modular and declarative programming interfaces across the wireless stack. As the existing wireless infrastructure suffers from the closely coupled hardware, OpenRadio aims to decouple the protocol definition from the hardware and provides a software abstraction layer to enable programmability at the MAC and physical layer. The main idea of Open-

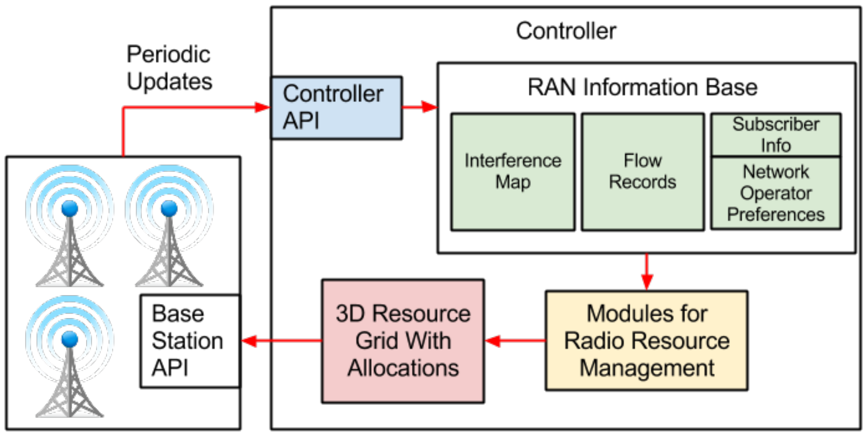


Figure 2.10: SoftRAN architecture [232].

Radio is to decompose wireless protocols into the processing plane and decision plane where the processing plane includes actions and the decision plane includes rules. Owing to the generality in its design, OpenRadio can be applied to both the cellular and the WLAN environment.

The existing cellular infrastructure has been criticized as being expensive and inflexible, suffering from complex control plane protocols and vendor-specific configuration interfaces [230, 234]. To simplify the management of cellular networks, the pioneering project CellSDN [230] proposes an SDN design for the cellular core networks. Given the challenges for applying SDN to cellular environment such as user mobility and real-time adaptation, CellSDN architecture includes a set of necessary extensions, as illustrated in Figure 2.11. On the proposed controller extension, policies of subscriber attributes are translated into switch rules for matching packet headers. CellSDN utilizes local cell agents to alleviate the scalability issue of central controlling by performing localized actions guided by the control platform.

Based on the high-level design of CellSDN, SoftCell [234] is a scalable architecture design to support fine-grained policies in cellular core networks. In SoftCell, the controller handles low-level details such as switch location and network identifier. Based on subscriber attributes and applications, SoftCell adopts a set of service policies to form a high level abstraction. The service policies include priority, service action, and predicates. SoftCell aims to improve scalability by extending both the control plane and the data plane. For the control plane, SoftCell uses local software agents to cache packet classifiers and policy tags in order to reduce the load of

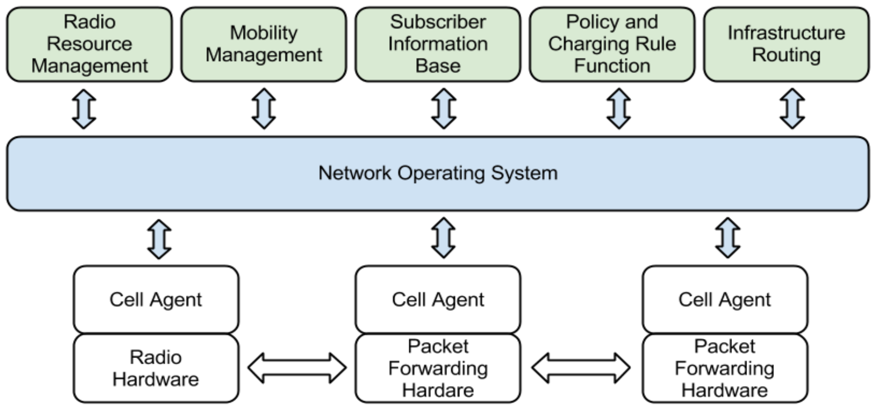

Radio Resource Management
Mobility Management
Subscriber Information Base
Policy and Charging Rule Function
Infrastructure Routing
Network Operating System
Cell Agent
Radio Hardware
Cell Agent
Packet Forwarding Hardare
Cell Agent
Packet Forwarding Hardware

Figure 2.11: CellSDN with local agents and extensions [230].

the main controller. For the data plane, SoftCell pushes the packet classification to access switches and applies multi-dimensional forwarding rule aggregation to minimize the state in the core network.

### *Wireless Local Area Networks*

OpenRoads [231] is the first SDN solution deployed on a campus that uses OpenFlow and virtulization to decouple mobility from the physical network and allow for multiple providers to control and configure the underlying infrastructure concurrently. The OpenRoads platform aims at enabling experimental research by using an open-source controller and extending it to control and capture wireless events such as host-AP association. The defining feature of OpenRoads is its openness - all the tools developed for OpenRoads are free and available under open-source licenses to encourage contributions from the community.

Odin [235] develops an SDN platform targeted at the enterprise wireless local area networks. By introducing a light virtual access point (LVAP) abstraction, Odin can virtualize the WLAN association state and separate the state from the physical access point to simplify the network management. The LVAP design enables the operators and developers to program and deploy WLAN services as network applications that run on top of Odin. For instance, Odin supports efficient handoff by using LVAP to avoid additional message exchange. The Odin mobility manager can monitor the receiver signal strength through local Odin agents to guide the selection of target AP for handoff. As illustrated in Figure 2.12, the key components

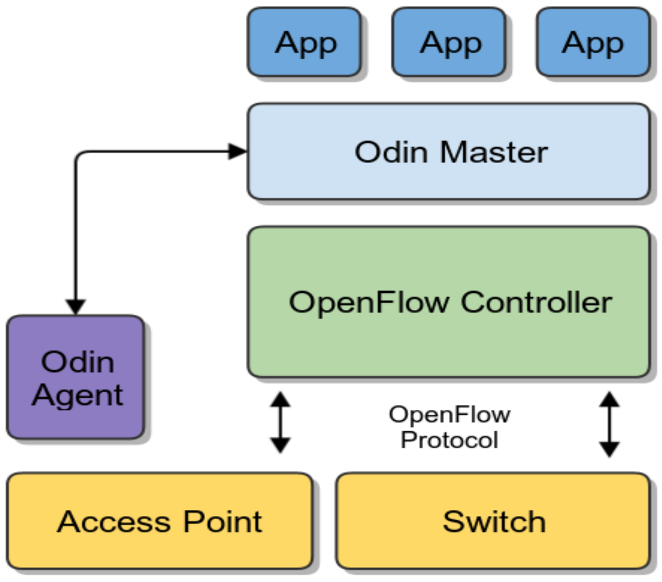


Figure 2.12: Odin system architecture [235].

of Odin such as Odin Master are built on top of the open-source OpenFlow controller Floodlight.

For edge access, OpenAPI [236] is a system proposal that aims to enhance efficient sharing of WLAN resources by virtualizing the last-mile access infrastructure via open APIs. OpenAPI introduces an algorithm to achieve efficient virtualization by leveraging the time elasticity of bulk data transfer applications and the spatial overlap of WiFi coverage. The proposed architecture specifies a set of interfaces for ISPs, content providers and end users to encourage collaboration of all the engaged parties: ISPs can improve the monetization of their infrastructure resources on a flow basis without revealing network internal details; the content providers can enhance their business model by selectively tuning the service quality for different types of flows; and the end users can adjust the virtualization degree to meet the dedicated needs.

***Discussion***

The proposals presented are good examples to demonstrate the potential of SDN to enhance mobile network architectures and resource management. In the search for novel approaches for mobile networks, it is worth recognizing that SDN is merely a tool for solving network-management problems. SDN neither dictates how that control should be designed nor solves any particular problem. Instead, it provides a platform to help address the longstanding problems in network and traffic management. The development of new solutions will require innovation on multiple timescales, from long-term bold visions and short-term creative problem solving. In this respect, we outline design challenges and requirements in mobile networks.

- *Mobility and roaming* – The mobility of users leads to roaming between networks and potentially across different access technologies such as 4G and WiFi. The dynamic change of locations adds complexity to manage and steer the mobile traffic across different access networks.
- *Multi-access and multi-operator* – The operational environment consists of different technologies and operators. It is leading to a complex negotiation process, privacy concerns, and potential conflicting policy and QoS requirements.
- *Monitoring overhead* – The OpenFlow-based monitoring schemes suffer from limitation in terms of high overhead and incomplete sample information. For instance, FleXam [237] provides a good example of this problem.
- *Deployment concern* – Although a defining feature of SDN is openness, solutions need to face the challenge of backward compatibility and interoperability as operators have to maintain different generations of technologies such as GSM, 3G and 4G.

## 2.5 Summary

This chapter presents the state of the art of recent solutions for mobile traffic offloading. By illustrating the major challenges in mobile networks, we discuss the values of mobile traffic offloading and highlight the importance of energy awareness in solution design. We present the latest development in software-defined networking (SDN) and several SDN-based schemes for mobile networks.

One of the key goals of this dissertation is to investigate mobile traffic offloading through experimental measurement and system design. Therefore, we mainly focus on system-level solutions. The observations based on literature study presented in this chapter lead us to develop collaborative, energy-aware, and software-defined solutions that will be presented in the next chapters.

# Chapter 3

# Network-Assisted Offloading (NAO)

This chapter presents the framework of Network-Assisted Offloading (NAO) for enabling efficient IP traffic offloading. Section 3.1 describes the motivation and measurement studies that drive our traffic offloading design. Section 3.2 illustrates the NAO architecture and protocol suite. Section 3.3 presents our system implementation and experiments in operational networks. Section 3.4 discusses existing standard-track mechanisms and compares our proposals against 3GPP-based solutions. Section 3.5 summarizes the chapter.

## 3.1 Requirement and Measurement Studies

Nowadays, 3GPP packet core network architecture and live network deployment tend to aggregate IP traffic at Gateway GPRS Support Nodes (GGSNs) or Packet Data Network Gateway (P-GWs) with relatively few sites. The IP traffic forwarding capacity of these aggregated gateway nodes is challenged by the phenomenal mobile traffic growth in recent years. Since the increased capacity investment requirements can be hard to meet due to the declining average revenue per user, mobile network operators are evaluating solutions to offload bulk IP traffic to alternative access technologies that are cheaper to deploy, such as WiFi.

To address the research question *'What are the key issues in mobile traffic offloading'*, it is worth assessing the traffic characteristics, buffering, and congestion in mobile networks to understand *'Why do we need to offload mobile traffic'* and *'What are the implications for mobile users and network operators'*. Through the WiBrA project (2010–2012), we carried

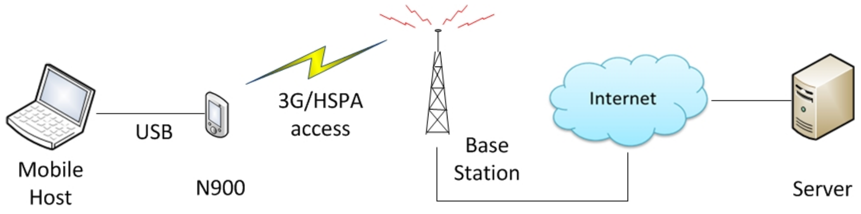


Figure 3.1: Test setup in the operational mobile network.

out a series of experimental measurements and simulation studies that covered mobile access networks [20, 21, 22]. The insights from those studies enhance our understanding of what the problems in existing mobile networks are, where the bottleneck is, and what to consider in designing IP traffic offloading schemes.

***Measurement Study***

As mobile broadband access becomes a popular choice for Internet access, the increased amount of TCP-based traffic has posed challenges to real-time media flows such as interactive video and audio. Volatile TCP flows such as web traffic are bursty, which can cause transient queues to appear and vanish rapidly at bottleneck routers. These transient queues introduce delay spikes that result in harmful jitters for the competing traffic such as real-time media flows. Meanwhile, long-lived TCP connections such as software updates cause standing queues to form and hence increase the one-way delay for competing traffic sharing the same bottleneck. Moreover, efforts to increase the initial window of TCP to ten segments have been made in recent years [238, 239]. Such increase together with a large number of parallel TCP connections have created a competitive environment for real-time media traffic.

To gain insights into the effect of competing TCP traffic on real-time media flows, we conducted experiments in an operational mobile network in Finland. The test system comprises a mobile host and a fixed server as presented in Figure 3.1. The mobile host was a laptop connected to a Nokia N900 smartphone to gain access to the high-speed mobile network. The fixed server was located in our laboratory with public IP address. All the test traffic was captured using tcpdump on both the mobile host and the fixed server. We carefully synchronized the clocks of the mobile host and the server before each test suite by using Network Time Protocol (NTP) to enable the clocks to be slowly adjusted towards almost equal rates. This enabled us to measure the one-way delay of each media packet

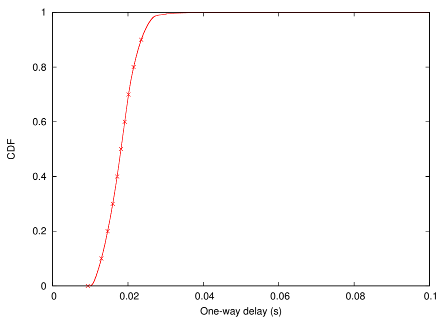


Figure 3.2: CDF of one-way delay for audio only workload of 15 seconds, 50 replications [20].

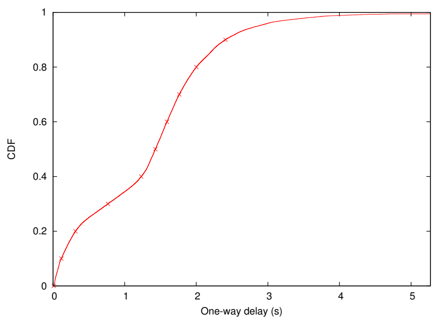


Figure 3.3: CDF of one-way delay for an audio flow with a competing bulk TCP connection, 50 replications [20].

with reasonable accuracy by taking the difference in timestamps obtained from the tcpdump logs of each end.

We used emulated traffic flows to gain full control of the workloads. The direction of traffic was from the fixed server to the mobile host. We used constant bit-rate (CBR) flows to emulate real-time audio traffic. For TCP traffic, we investigated two major workloads that roughly mimic two typical scenarios where TCP flows compete with the real-time audio traffic: 1) Software update (bulk TCP flow) during a voice call and 2) Web browsing (short TCP flows) when a voice call is ongoing.

As a baseline, Figure 3.2 shows the cumulative distribution function (CDF) of the one-way delay for an audio flow of 15 seconds with 50 replications. During the observed period in the operational mobile network, the loss rate was only 0.05%. The delays remained below 40 ms except for a handful of packets. The median and the maximum one-way delay were 18.0 ms and 70.4 ms, respectively.

To highlight the effect of competing traffic on latency, Figure 3.3 presents the CDF of the one-way delay for real-time media (Audio) traffic during a bulk TCP transfer. Figure 3.4 shows the CDF of the one-way delay for audio traffic when different numbers of TCP connections with different initial window (IW) sizes were in use. We ran 50 replications for each test case with different combinations of parameter values. The bit-rate of emulated audio flows is 16 kbps, which equals an IP packet of 80 bytes transmitted every 20 ms. For the competing bulk TCP workload, an emulated audio flow started first and then a bulk TCP flow started to transfer 28 MB of data. The start time of bulk TCP was distributed uniformly between 10 to 12 seconds after the start of the audio flow. The number of parallel short

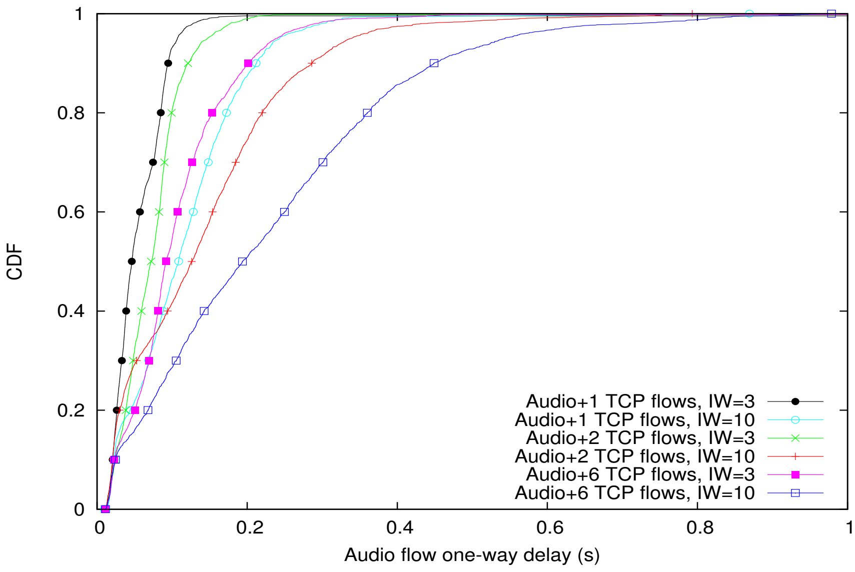


Figure 3.4: CDF of one-way delay for an audio flow competing with one, two, and six TCP flows, 50 replications [20].

TCP flows can be one, two and six. The total size of the TCP flows is 372 KB. In all cases, audio traffic started first and the starting time of TCP flows was distributed uniformly between 10 to 12 seconds after the start of the audio flow. The audio traffic lasted long enough to cover the whole duration of TCP flows.

When competing with a bulk TCP flow as shown in Figure 3.3, interactive audio was almost impossible because the one-way delays during the TCP transfer were excessively high. The 25th percentile of the one-way delay was already 0.5 seconds and the median was 1.42 seconds. We confirmed from the traces that deep buffering is the main cause for the delay increase, as soon after the bulk TCP transfer started the one-way delay increased and remained around 1.5-2.5 seconds consistently for the duration of the TCP transfer. Such a delay increase was not present in our audio-only tests. We noted that there were few values in the highest end, which, however, might be due to wireless network phenomena on top of the deep buffering.

As shown in Figure 3.4, the one-way delay with one competing short TCP flow using an initial window of three segments (IW=3) was reasonably low, which allows smooth packet delivery for interactive media. Increasing the number of TCP connections from one to two caused only a moderate increase in the end-to-end delay. Increasing the TCP connections to six resulted in relatively larger one-way delays. In all cases with competing TCP traffic using IW=3, the one-way delay remained below 150ms all the way up to the 75th percentile. In the cases of TCP flows using IW=10, the one-way delay was notably higher compared with the values in the corresponding IW=3 cases. The median one-way delay with six competing TCP flows using IW=10 approached 200ms. The delay values remained below 150ms for one and two competing TCP flows with IW=10.

Real-time media flows are typically compressed by a codec which tries to retain human observable properties of the original content while removing redundant information. The sporadic losses in transmission are usually concealed by the codecs. However, when more losses occur consecutively, the quality deteriorates and distortions become noticeable. To absorb jitter that occurs in the packet transmission, a jitter buffer is utilized between the receiving codec and the network. Because the media playback is time bounded, the codec requires the data to be delivered in time. If a sudden increase of delay occurs in the network, the media packet may not arrive in time for the playback and the codec will discard the packet when its delay exceeds the pre-defined threshold determined by the jitter buffer size.

In order to explore the transient effect of the delay jitter on the media flows in mobile networks, we introduced a jitter filter to mimic the receiving codec behavior in dropping late arriving media flow packets, referred to as 'delay-based loss'. The delay-based losses are flagged when one-way delay of the packet exceeds the 'base delay' plus the pre-defined jitter buffer size. The 'base delay' is calculated from the perceived minimum delay over the period of two seconds prior to the arrival of the TCP flows. We intentionally chose to use minimum delay as base delay in order to report the worst-case behavior.

We highlight the loss rate with different jitter buffer sizes in Figure 3.5 for a media flow (Audio) competing with one, two, and six short TCP flows with IW=3 and IW=10. We conducted 50 replications for each test case. The loss rate is determined by combining the delay-based losses and the pure losses which were caused by either network congestion or link errors. We plotted the median value of loss rate and 25th and 75th percentile in the figure.

According to the measurement data, larger jitter buffer sizes can lower the loss rate especially for IW=3. Using IW=10 in TCP increases the loss rate dramatically to nearly 90% with small jitter buffer sizes. Meanwhile, even with IW=3, a large number of parallel TCP connections can cause a significant number of losses.

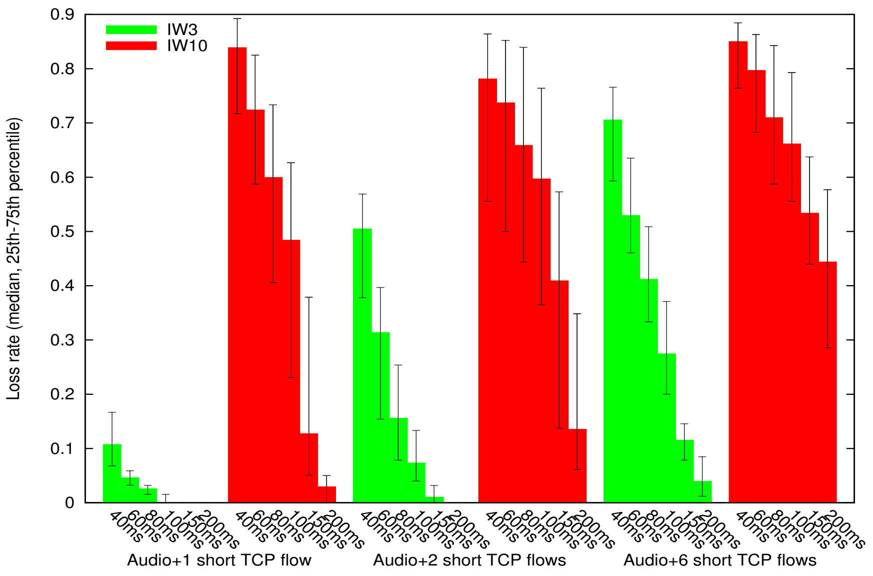


Figure 3.5: Loss rate (delay based + pure losses) with different jitter buffer sizes [20].

### *Simulations*

On the network side, a phenomenon called bufferbloat [240] has made the problem in mobile networks more challenging. Typically, buffers are deployed in the network for multiple reasons: 1) to smooth the transient traffic spikes, 2) to enable packet lossless handovers, and 3) to avoid missing packets due to wireless errors. However, excessive buffering is harmful [241], particularly for interactive traffic and protocols that are designed to cope with packet losses.

Because of bufferbloat, devices in the network can end up buffering an enormous amount of traffic such as the initial windows of all parallel web responses. The recent effort to increase the initial window of TCP from three to ten segments can make the situation even worse. Our measurement also confirmed the existence of such deep buffering in the operational network [20]. Active queue management (AQM) and its most prominent representative Random Early Detection (RED) [242] is often proposed as a solution to the bufferbloat.

In mobile networks, the buffering/queuing system at the last-hop forwarding nodes tend to have limited aggregated traffic generated by a single

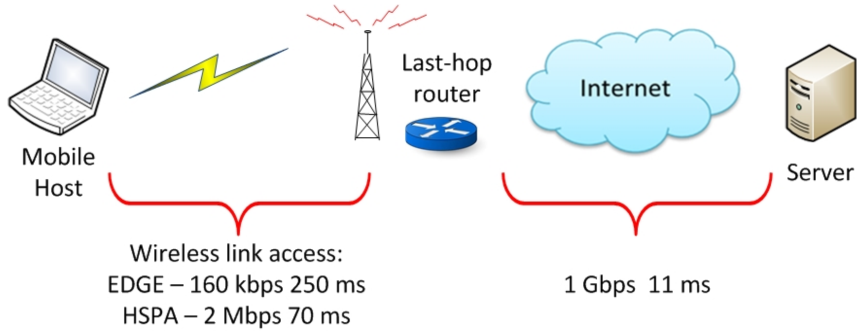


Figure 3.6: Simulation topology.

user or a few users. With the main goal of improving RED for such limited aggregated traffic [21], we conducted simulation studies in an environment mimicking a mobile broadband access. Because it is difficult to access the operator resources, the simulation studies help us to investigate performance issues resulting from the competing traffic. Figure 3.6 shows the simulation topology similar to the setting for our live network measurement.

Through simulation, we examined the competing Web traffic and real-time traffic in mobile access when AQM schemes were applied to the bottleneck router. Table 3.1 shows the results with TCP competing traffic and buffer size of two times the bandwidth-delay product (2BDP) in an HSPA link. Two types of TCP workloads were included: 1) one background bulk TCP flow with a burst of later starting TCP flows and 2) a burst of n TCP flows with a burst of n later starting TCP flows. The background bulk TCP flow and the first n TCP flows started at zero seconds. The background bulk TCP flow was left to settle into the congestion avoidance mode of operation before the later TCP burst started. Each burst consists of n=1, 2, 6, or 18 TCP flows that start simultaneously. The total size of the TCP burst flows was 360 kB divided equally between the flows in the given burst. The CBR payload rate was set to 64 kbps for HSPA. The CBR start time was distributed to occur 0–200 ms earlier than the later starting TCP burst.

As shown in Table 3.1, when competing with a bulk TCP and TCP short flows under first-in first-out (FIFO) with buffer size of 2BDP, all packets of the CBR flow were above one-way delay of 200 ms, exhibiting the worst performance. The situation was improved when the aggregated router applied AQM/RED. In the case of bulk TCP and 1 TCP flow, 92% of the CBR packets were below 200 ms one-way delay. However, when there

Table 3.1: Percentage of CBR packets with one-way delay below 200ms over HSPA link [21].

| | Bulk TCP+n TCP Flows | | | | n + n TCP Flows | | | |
|---|---|---|---|---|---|---|---|---|
| n | 1 | 2 | 6 | 18 | 1 | 2 | 6 | 18 |
| FIFO+2BDP | 0% | 0% | 0% | 0% | 82% | 62% | 30% | 26% |
| RED+2BDP | 92% | 75% | 38% | 28% | 82% | 62% | 31% | 28% |

were several parallel flows competing, the performance of real-time traffic will degrade even when RED was applied. In the case of bulk TCP + 18 TCP flows, merely 28% of the CBR packets were below 200 ms one-way delay.

***Observations***

Our measurement studies illustrate how interactive media flows are affected by concurrent TCP transmissions that share the same aggregation link in high-speed mobile access networks. In the conducted experiments, we experienced very few packet losses during the observed period in an operational mobile network. The impact of competing traffic is hence reflected mainly by delay and changes in the delay.

We notice that even a moderate number of parallel TCP connections that are typically used for carrying Web page responses can cause irreparable harm for a concurrent interactive media flow sharing the same bottleneck link. With a competing bulk TCP transfer, the media stream becomes almost unusable for interactive purposes. In our measurement, the worst competing effect toward real-time media flow resulted from the burst of packets during the initial TCP window transmission. Compared with the initial window of three segments, the initial window of ten segments can worsen the situation.

Concerning the deep buffering/bufferbloat, it is worth noting that although AQM is regarded as a viable candidate, it is challenging to realize in practice. The access network gateways/routers, which are typical bottlenecks, lack support for AQM/RED. Even if available, RED does not work with the default settings as it is too gentle to handle fast changes due to TCP slow start when the links are subject to limited aggregate traffic [21]. Since tuning of the RED parameters requires modifications on the intermediate network nodes, it is not deployable in the short run on a large scale even if RED itself is supported by the devices.

Given the radical growth of mobile traffic, the effort to increase the

TCP initial window to ten, and the trend of deep buffering on the mobile aggregate gateways, congestion and competing traffic effects are bound to occur to degrade the performance of mobile services, especially for interactive applications such as voice, video and online gaming. Because mobile gateways such as GGSN or P-GW tend to heavily aggregate traffic, the IP traffic forwarding capacity of these gateway nodes might not sustain the traffic growth to meet the requirement of service quality.

Our key observation is that the existing problems in mobile networks are accumulated over the years with many stakeholders and factors intertwiningly involved. The collective impact is hence complex, appearing in various forms with different symptoms. Instead of trying various mitigation schemes to fix problems within the incumbent mobile access infrastructure which often entails high costs and complexity, we can consider other promising alternatives, such as to offload bulk data traffic to other wireless channels, e.g., WiFi, to alleviate the pressure on the overloaded mobile access. The latest trend to equip mobile devices with an additional WiFi interface and the fast growth of WiFi hotspot deployment in urban areas also support the vision. This observation motivates us to investigate traffic offloading design as a cost-effective mechanism for mobile network operators to mitigate the challenge of traffic growth in the upcoming years [243]. For mobile users, offloading bulk traffic to alternative accesses can also bring potential benefits in terms of performance and cost saving [5, 6].

Since we focus on traffic offloading in this dissertation, we only cover a selected set of measurement results that motivate our offloading design. A detailed discussion including how to improve AQM/RED can be found in [20, 21]. Our experiment data from the operational network is available at: `http://www.cs.helsinki.fi/group/wibra/pam2013-data/`

## 3.2 NAO Framework Design

Network-Assisted Offloading (NAO) is a framework that utilizes network assistance to achieve efficient IP traffic offloading. It enables mobile network operators to actively trigger and guide the traffic offloading through dedicated protocols.

Figure 3.7 illustrates a typical offloading scenario of NAO to push network-driven offloading policies to mobile hosts using Router Advertisement (RA) [25, 76]. We assume mobile hosts are equipped with cellular and WiFi interfaces, which is a common setting for modern handset devices.

We consider two major cases in mobile traffic offloading: 1) to complement cellular radio coverage and access capacity with a cheaper radio

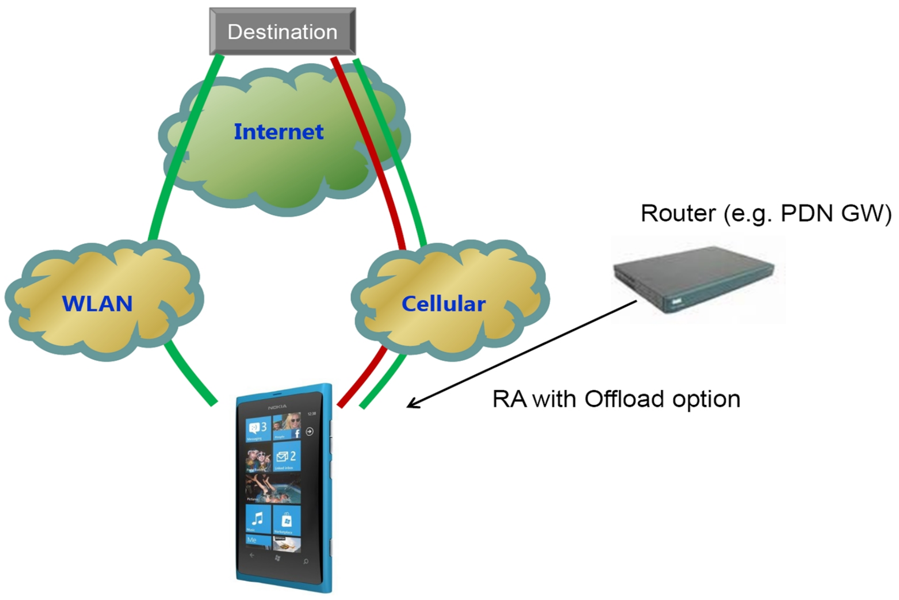


Figure 3.7: NAO offloading scenario.

technology in a dense hotspot area, but still route the traffic through the operator's packet core. 2) Bypass the operator's packet core, cellular access, and possibly backhaul completely to maximize savings. Although NAO targets the latter case, the proposed framework could apply to both.

### 3.2.1 Design principles

Given the pressure of traffic growth affecting radio access, backhaul, and network packet core, IP traffic offloading has been a lively topic in both standard and product development. Specifically, 3GPP has put considerable efforts in standardizing IP traffic offloading solutions for the Evolved Packet Core (EPC) in recent years. The 3GPP approaches, as discussed in Section 2.2.2, rely on tight cellular operator control and integration to 3GPP network architecture, demanding 3GPP-specific modifications for user equipment. We identify the issues of the existing network-controlled offloading approaches as follows:

1. Heavy system-specific standardization – This inevitably postpones deployment and ties the solution too tightly to a specific architecture.

2. The micromanagement of the offloading policies – The flow granularity in mobile networks can lead to unnecessary management burden,

particularly on the heavy-loaded aggregate gateways. Moreover, a reliable identification of flows is challenging due to encryption.

3. Requirement of 3GPP-specific functionality in user equipment – We believe that solutions should only be at the IP level and therefore independent of the access technology.

We advocate that IP offloading should be considered just a 'normal' IP routing and next-hop selection issue [244] with minimal management overhead. To guide the design of the NAO framework, we derive a list of principles based on literature studies [13] and measurement experience [20]:

- *Internet layer generic design* – As long as the Internet connectivity is guaranteed and the few required specific operator services can be reached, the IP traffic offloading scheme should be realized just as an IP-level routing and source address selection so that virtually every IP stack can support offloading without access-specific enhancements.
- *Efficient and flexible network assistance* – Given that the solutions requiring interface scanning may lead to extra energy cost [92], it is advisable to take advantage of network assistance to trigger the scanning on demand and thereby avoid the excessive energy consumption on resource constrained mobile devices. Meanwhile, instead of forcing tight network control, network assistance should allow both users and the network side to initiate offloading. This improves the flexibility and efficiency as traffic offloading can be driven in a timely manner according to demands. Specifically, the network side decision can reference the gateway nodes such as GGSNs or P-GWs that have an overview of the aggregated workloads in the mobile networks.
- *Lightweight and secure* – For deployability, the offloading solution should introduce minimal impact on infrastructure and end user devices. On user equipment, the standard IP stack supporting IPv4 and IPv6 should suffice. At the same time, the communication channel between the network and user equipment for managing traffic offloading must be both secure and trusted. For example, using cellular access to push offloading policies can be considered a secure option compared with using other open wireless accesses.

### 3.2.2 NAO framework

Figure 3.8 shows an overview of the NAO framework comprising three major parts: network side assistance, signaling protocols, and host operations.

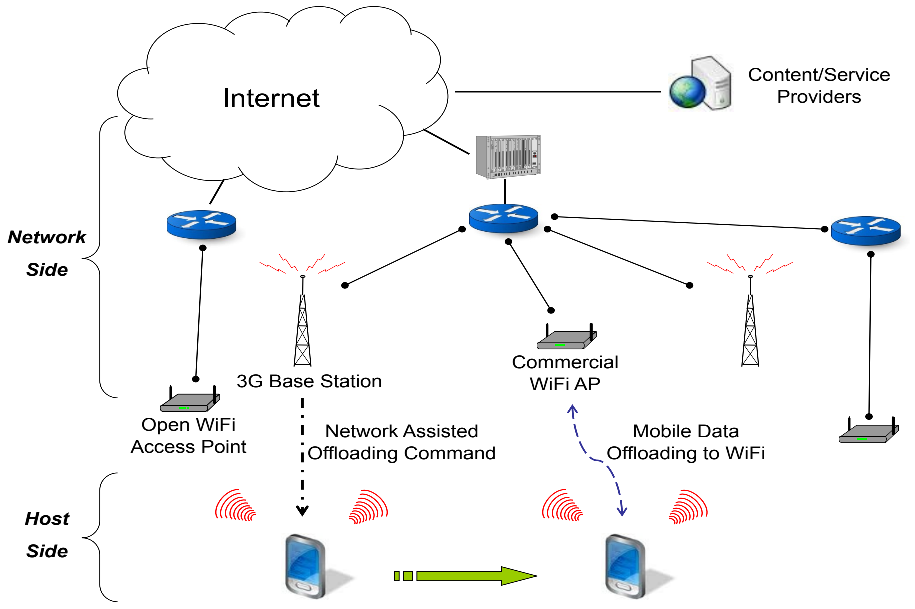

Internet
Content/Service Providers
Network Side
Host Side
3G Base Station
Commercial WiFi AP
Open WiFi Access Point
Network Assisted Offloading Command
Mobile Data Offloading to WiFi

Figure 3.8: NAO framework overview [23].

First, the network side assistance includes offloading preparation and decision supported by the network. Second, the signaling protocols serve as the bridge between the network side and hosts to deliver offloading commands and user requests. Third, host operations are necessary modules / software extensions deployed on mobile devices for fulfilling the offloading procedure.

NAO supports two offloading modes for enhancing flexibility, namely host-driven mode and network-driven mode. The host-driven mode allows end users to trigger traffic offloading according to their demand and preference by using the DHCPv6 route options [248]. The network-driven mode enables operators to actively push offloading policy by using a standardized Neighbor Discovery option [76] and our own protocol extension [25]. In both modes, once the offloading decision is made by the network side based on context such as network condition and user location, the dedicated gateway deploying NAO will send offloading commands to the targeted user equipment (UE) via a DHCPv6 reply message [249] or a router advertisement [250], for host-driven mode or network-driven mode, respectively. Upon receiving the offloading command, a NAO-compatible UE will carry out IP traffic offloading by selecting a proper networking interface for data transmission according to the indication.

In NAO, cellular access is chosen as the commanding channel to guide traffic offloading and deliver Internet layer offloading policies. By following the deployment guidance [245], we utilize proposals in IETF and combine them with our own extension to build the NAO signaling protocol suite.

The NAO signaling solutions have several aspects in common:

- They rely on IPv6 features when possible. No 3GPP-specific extensions are required.
- They primarily target UEs with cellular 3GPP access. The GGSN or PGW is used as the orchestra leader in the operator network.
- Cellular 3GPP radio is considered a *trusted* access and hence used to deliver offloading policies.
- Designed to benefit from multiple interfaces.

Solutions adopted in NAO aim to work on the Internet layer and rely only on the standard IETF defined TCP/IP protocol suite, without requiring any access technology specific knowledge. We consider Dynamic Host Configuration Protocol (DHCP) to be part of the Internet layer due to its traditionally tight integration with IP and host configuration. Our goal is to identify working solutions that do not break backward compatibility, work in the majority of the intended use cases, and would not require mobile devices to implement 3GPP-specific extensions. The emphasis of NAO is on IPv6, since we believe there is no room for sophisticated IPv4-based offloading solutions anymore, and the future is in IPv6 networking.

### 3.2.3 NAO protocol suite

To achieve efficient and flexible IP traffic offloading according to the demands of user and network operator, we utilize three Internet layer solutions. The first approach builds on top of DHCPv6 [249] which enables the host-driven mode. The second approach builds on top of IPv6 neighbor discovery (ND) protocol [250] and the third approach extends the second solution with IPv4 capabilities for supporting IP traffic offloading in the transition to the IPv6 [246, 247]. The second and third approaches together enable the network-driven mode.

#### ***Host-Driven Mode with DHCPv6***

We utilize the DHCPv6 route options [248] proposed in IETF to achieve the host-driven traffic offloading in our framework. This approach enables a NAO-enabled UE to obtain routing information for offloading purposes

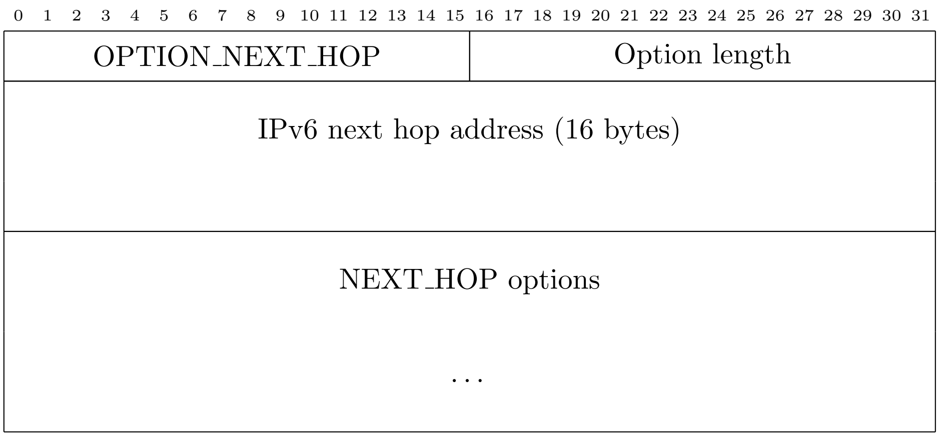


Figure 3.9: Next Hop (*NEXT_HOP*) option format.

from dedicated DHCPv6 servers that support the route options. We use two DHCPv6 route options in NAO: *NEXT_HOP* and *RT_PREFIX*. The message formats for *NEXT_HOP* and *RT_PREFIX* are illustrated in Figure 3.9 and Figure 3.10, respectively.

Since each IPv6 route consists of a destination prefix, a next-hop address, and a preference value for the route, the *NEXT_HOP* option contains the next-hop address and the *RT_PREFIX* option contains prefixes and associated preference values. The option codes for the two route options are OPTION_NEXT_HOP and OPTION_RT_PREFIX (numeric values are to be assigned by IANA [1]).

The *NEXT_HOP* contains the following fields, as shown in Figure 3.9:

- Option code: Defined as OPTION_NEXT_HOP, numeric value to be determined by IANA.
- Option length: 16 bytes + the length of NEXT_HOP options.
- IPv6 next-hop address: 16 bytes in length, IPv6 address of the next-hop.
- NEXT_HOP options: Options that are associated with this Next Hop option, including but not limited to, zero, one or more RT_PREFIX options that specify prefixes reachable through the given next-hop.

[1] Internet Assigned Numbers Authority (IANA): `http://www.iana.org/`

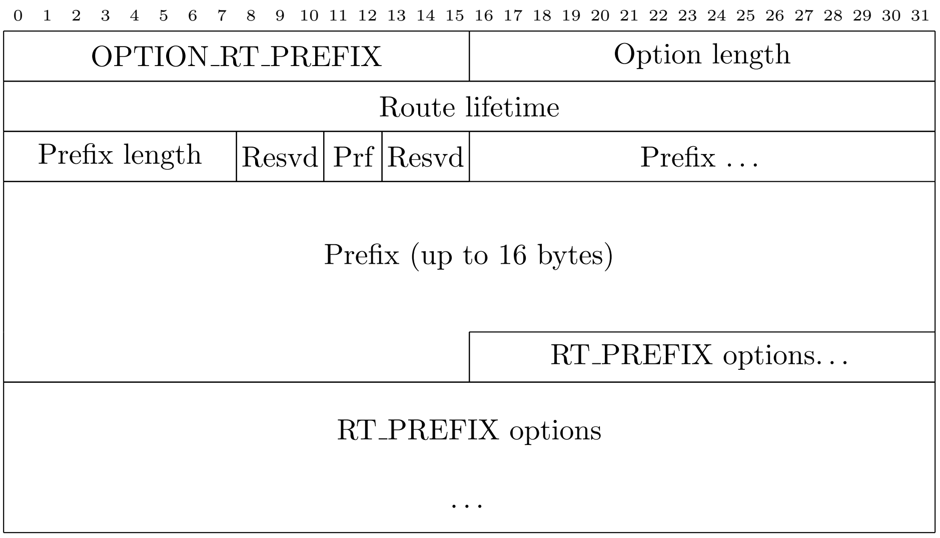


Figure 3.10: Route Prefix (*RT_PREFIX*) option format.

The *NEXT_HOP* option as shown in Figure 3.9 usually contains the IPv6 address of a specific next-hop router. The *NEXT_HOP* option also contains *RT_PREFIX* as its suboptions. For each next-hop address, there can be zero, one or more prefixes reachable via that next-hop.

The *RT_PREFIX* option contains the following fields as shown in Figure 3.10:

- Option code: Defined as OPTION_RT_PREFIX, numeric value to be determined by IANA.

- Option length: Length of RT_PREFIX option including all the sub-options.

- Route lifetime: 32-bit unsigned integer to specify the lifetime of the route information, expressed in seconds relative to the time the packet is sent. Value 0 means that the route is no longer valid and must be removed. Value 0xffffffff indicates infinity for the lifetime.

- Prefix length: 8-bit unsigned integer to represent the length of the IP prefix in bits. The value ranges from 0 to 128.

- Resvd: Reserved field. DHCP severs must set the value to zero and clients must ignore its content.

- Prf (Route Preference): Preference values are encoded as a 2-bit signed integer. It indicates whether to prefer the next-hop router associated with this prefix. Value '01' indicates high preference. Value '00' indicates medium preference and value '11' indicates low preference. Value '10' is reserved.
- Prefix: A variable size field (up to 16 bytes) for specifying an IPv6 prefix. The length of the field is defined by the prefix length. If the length is not divisible by 8, padding should be applied to round up to the nearest byte boundary. The padding must be zero.
- RT_PREFIX options: Extended options for this specific IPv6 prefix.

The RT_PREFIX option can convey routing information through an IPv6 prefix that represents a route destination. It can be used to configure a default route by specifying a prefix of "::/0" in the RT_PREFIX option which is included as a suboption in the NEXT_HOP option. The route preference value in RT_PREFIX is useful for multi-access scenarios where there can be multiple next-hop routers for a single prefix (e.g., one next-hop router located in WiFi access and another one in the cellular access). By setting the route preference value, network operators can guide UEs to select a proper next-hop router and thereby manage IP traffic between WiFi access or cellular access in an on-demand manner.

By relying on a DHCP-based approach, UEs must check the reachability of next-hop routers through standard mechanisms such as Neighbor Unreachability Detection (NUD) [250]. To avoid signaling overhead, DHCPv6 servers should not send route options to UEs that do not explicitly request the options.

In NAO, we utilize the route options for enabling host-driven IP traffic offloading. A UE can initiate traffic offloading on demand by sending a request message that includes the *NEXT_HOP* and *RT_PREFIX* options as part of the DHCPv6 Option Request Option (ORO) [249]. By receiving the request, a NAO-enabled server provides routing configuration information to the requesting UE according to pre-defined network policies. A typical use case is when a NAO-enabled UE connected to both WiFi and cellular accesses, a network operator can guide traffic offloading by installing a default route via the route options to point to a next-hop router in WiFi access. Once the UE learns this new default route, it will update its routing table and offload traffic to its WiFi access rather than using cellular access. To achieve a fine-grained control, route options also allow operators to set the route preference values for specific prefixes, so that the traffic matching those prefixes can be directed to desired access networks (either WiFi or cellular). In brief, NAO allows UEs to pull offloading policies using the DHCPv6 route options, thereby enabling the host-driven traffic offloading.

#### *Network-Driven Mode with Neighbor Discovery*

To enable network-driven mode in NAO, we adopt the IETF *"Default Router Preferences and More-Specific Routes"* [76] which extends IPv6 Neighbor Discovery protocol with a new Route Information Option (RIO) and preference flags in the Router Advertisement (RA) message header. The new extension allows operators to convey default router preferences and specific routes to UEs.

The new option and preference flags require changes to the Router Advertisement in the Neighbor Discovery protocol [250]. Figure 3.11 shows the format of RA with the router preference extension.

The major changes as shown in Figure 3.11 include:

- Prf: Default Router Preference, expressed in 2-bit signed integer. Value '01' indicates high preference. Value '00' indicates medium preference and value '11' indicates low preference. Value '10' is reserved. The preference value indicates whether to prefer this router over other default routers.
- Resvd: A 3-bit field which must be initialized to zero by the sender and must be ignored by the receiver.
- Options: The field for RA options such as Route Information Option.

The regular fields in RA defined by Neighbor Discovery protocol [250, 251, 252] include:

- Type: This field indicates the type of ICMP message. The value for RA is 134.
- Code: This field depends on the message type for additional level of message granularity. The value for RA is 0.
- Checksum: The ICMP checksum [252].
- Cur Hop Limit: 8-bit unsigned integer. It contains the value that should be placed in the Hop Count field of the IP header for outgoing IP packets.
- M: 1-bit flag field for managed address configuration. If this flag is set, it indicates that addresses are available via DHCPv6 [249].

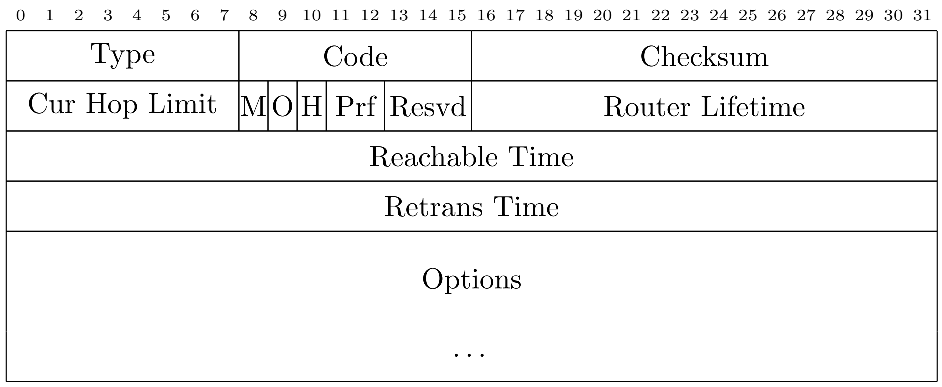


Figure 3.11: Router Advertisement format with preference extension.

- O: 1-bit flag field for other configuration. If this flag is set, it indicates that other configuration information is available via DHCPv6, such as DNS-related information.
- H: 1-bit Home Agent field for supporting Mobile IPv6 [251]. The 'H' bit is set in a RA message to indicate that the router sending this RA is functioning as a Mobile IPv6 home agent on this link.
- Router Lifetime: 16-bit unsigned integer specifying the lifetime associated with the default router in seconds. This field is used only to indicate the usefulness of this router as a default router. It does not apply to fields in other options. A lifetime value of zero indicates that the router is not a default router and should not appear on the receiver's default router list.
- Reachable Time: 32-bit unsigned integer, indicating the time in milliseconds for assuming a neighbor is reachable since the most recent reachability confirmation received.
- Retrans Timer: 32-bit unsigned integer, indicating the time in milliseconds between retransmitted Neighbor Solicitation messages.

The default router preference as presented in Figure 3.11 allows for a simple three step prioritization of default routers in the host's default router list (*LOW, MEDIUM - the default, and HIGH*). In addition to the preference value, the 'Options' field can contain a Route Information Option which provides extra configuration context for specific routes.

The Route Information Option contains the following fields, as shown in Figure 3.12:

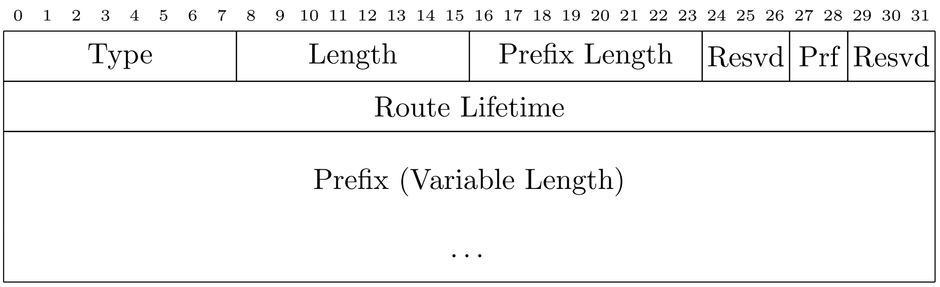


Figure 3.12: Route Information Option message format.

- Type: This field indicates the type of RA option. The value for RIO is 24.
- Length: 8-bit unsigned integer in units of 8 bytes for indicating the length of the option including Type and Length fields.
- Prefix Length: 8-bit unsigned integer, indicating the number of valid bits in Prefix, ranging from 0 to 128.
- Prf: Route Preference, 2-bit signed integer for indicating whether to prefer the router associated with this prefix over other routers when the same prefix matches multiple routers.
- Resvd: Unused field. Senders must initialize this field to zero and receivers must ignore the values in this field.
- Route Lifetime: 32-bit unsigned integer, indicating the length of lifetime in seconds for a prefix. The value of 0xffffffff represents infinity.
- Prefix: This field contains an IPv6 address or an IPv6 prefix whose length is determined by the Prefix length.

With the new RIO, a router can indicate specific IPv6 routes that the router wants to serve as the first-hop. Similar to the DHCPv6 approach, the Route Preference in RIO can be used to provide fine-grained routing management in multi-access environments. For example, where a single prefix on a multi-interfaced UE matches two routers, one located in WiFi access and the other one in the cellular access, network operators can guide the UE by prioritizing the Route Preference for the WiFi access router and thereby direct IP traffic generated by the UE to the preferred WiFi network.

The main reason for adopting RA router preference and RIO in our framework is to take advantages of an existing specification [76] on top of the Neighbor Discovery protocol. Several mainstream operating systems already implement RFC4191, including Linux, BSD variants and Microsoft Windows starting from XP. The network interfaces accepting RFC4191 extensions can be specified using host side configuration. In addition, RA is regarded as a stable mechanism for router-to-host communication. The only consideration is to limit the number of interfaces accepting RFC4191 extension to one interface that is both trusted and centrally managed by the operator. In our case the 3GPP cellular connection fulfills these requirements. In brief, the RA extensions in NAO allow network operators to actively push offloading policies to UEs from the access routers, thereby enabling the network-driven traffic offloading.

#### *Network-Driven Mode with IPv4 Support*

To overcome the limitations of the IPv6-only approach specified in RFC 4191, we proposed two Neighbor Discovery options for NAO to support IPv4 traffic offloading [25]. The new options enable access routers to communicate the IPv4 default gateway address and more specific IPv4 routes to UEs. The operation logic for traffic offloading is similar to the IPv6 approach [76].

The new option is named Offload and used in the Router Advertisement. Figure 3.13 shows the message format of the Offload option.

- Type: Defined as OPTION_OFFLOAD, numeric value to be determined by IANA.

- Length: 8-bit unsigned integer specifying the length of the option in units of 8 bytes, which includes the Type and Length fields. If there is no sub-option, the value must be 2.

- D: 1-bit flag field for indicating the willingness of the router to serve as a gateway for IPv4 traffic. If 'D' flag is set to 1, the router explicitly indicates it is not willing to serve as a gateway for IPv4 traffic when there are other usable gateways accessible from the the same or other available interfaces. If 'D' is set to 0, the router indicates no preference as to its willingness to serve as a IPv4 traffic gateway.

- Reserved: Unused field, which must be initialized to zero by the sender and its content must be ignored by the receiver.

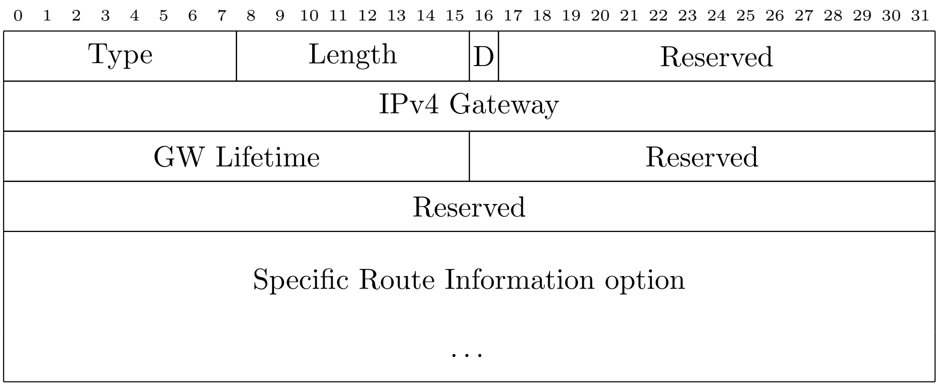


Figure 3.13: Offload option in router advertisement.

- IPv4 Gateway: The IPv4 address of a gateway's interface. This address must belong to the same interface that originates the RA containing the Offload option.

- GW Lifetime: 16-bit unsigned integer, indicating the lifetime in seconds for the gateway conveyed in this Offload option. Its value should be smaller than or equal to that of Router Lifetime, which is in the ICMP header of the same Router Advertisement.

- Specific Route Information option: Optional information for specific routes that are used for IPv4 traffic offloading.

The new Offload option is used in Router Advertisement (RA) to convey routing information of an IPv4 gateway for traffic offloading. To avoid misconfiguration, we only allow one Offload option in a single RA. The "D" flag, IPv4 Gateway, and GW Lifetime fields can be used by network operators to instruct UEs to offload their IPv4 traffic. For instance, a default gateway for IPv4 traffic can be added, updated, or deleted depending on the "D" flag, GW Lifetime and existing routing state on a UE. When the "D" flag is set to 1, the existing default gateway matching the one contained in the Offload option should be removed if there are other usable gateways through the same or other interfaces. In this case, the following IPv4 traffic can be redirected to another gateway preferred by the network operators for traffic offloading.

To enable IPv4 traffic offloading, we defined a Specific Route Information (SRI) option that can be used as a sub-option for the Offload option. Figure 3.14 presents the message format of the SRI option.

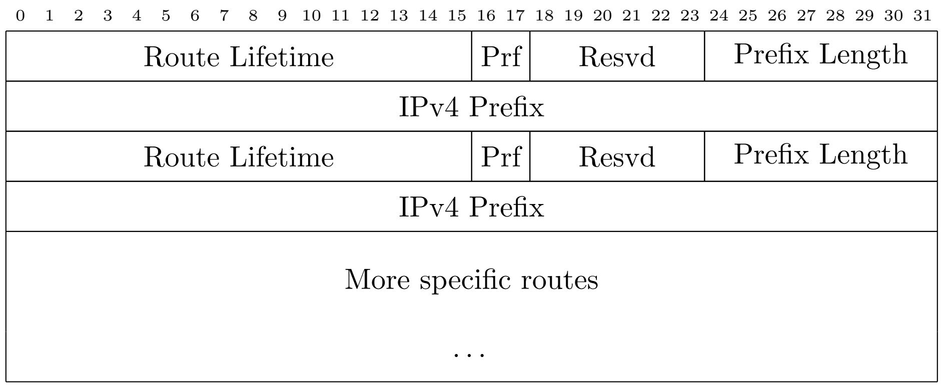


Figure 3.14: Specific Route Information option message format.

- Route Lifetime: 16-bit unsigned integer indicating the length of time in seconds for the corresponding IPv4 prefix. Its value should be smaller than or equal to that of GW Lifetime contained in the same Offload option.

- Prf (Route Preference): 2-bit signed integer indicating whether to prefer the router associated with this prefix over other routers when the same prefix matches different routers. The values are encoded as follows: Value ‘01’ indicates high preference. Value ‘00’ indicates medium preference and value ‘11’ indicates low preference. Value ‘10’ is reserved.

- Resvd (Reserved): 6-bit unused field, which must be initialized to zero by the sender and must be ignored by the receiver.

- Prefix Length: 8-bit unsigned integer indicating the number of bits in the IPv4 Prefix field that are valid.

- IPv4 Prefix: 32-bit field, containing the IPv4 prefix for a specific route.

The Specific Route Information option must be included in the same Offload option for conveying IPv4 traffic offloading information. To support traffic offloading, a host that receives those options needs to maintain a set of states including the IPv4 Gateway, IPv4 Prefix, Prefix Length, Route Preference, and Route Lifetime. For example, when the ‘D’ flag is set to 0 in the Offload option, then the advertised SRI can be processed by the host if there is no duplicated IPv4 prefix matching the advertised IPv4 prefix and the Route Lifetime is valid. If there is a matching prefix, the corresponding specific route will be updated or removed according to the Route Preference and Route Lifetime. In particular, the Route Lifetime determines whether the route shall be removed or updated depending on the existing routing state of the host. If the Route Lifetime is set to 0, any matched IPv4 prefix and corresponding gateways must be removed. If the Route Lifetime is valid, the Route Preference will be used to further determine whether to prefer the advertised gateway over other options. Meanwhile, when the 'D' flag is set to 1 in the Offload option, any existing specific routes with the next-hop router matching the advertised IPv4 Gateway must be removed.

Our IPv4 offloading solution extends the Neighbor Discovery protocol for hosts without DHCP support and also works in networks that do not utilize DHCP. It is intended to be used in transition towards IPv6, during which time multi-interfaced hosts are likely to have network interfaces with IPv4-only capability. A common scenario where coexistence of IPv4 and IPv6 network interfaces is expected to occur is when a smartphone has an IPv6-enabled cellular connection and an IPv4-only WiFi connection active at the same time. By conveying gateway and specific route information through the proposed options, our solution enables network operators to offload IPv4 traffic from cellular access to WiFi for such a multi-interfaced dual-stack capable host. In brief, our IPv4 offloading proposal for NAO allows network operators to actively push offloading policies to UEs from the access routers, thereby enabling the network-driven IPv4 traffic offloading.

## 3.3 Implementation and Experiments

We validate the NAO framework by developing prototype systems and testing them in live cellular and WiFi networks. The main goal is to illustrate how to conduct mobile traffic offloading at the IP level. We implement the NAO protocol suite and show that the proposed offloading solutions are both feasible and lightweight to deploy on the network side and on mobile devices.

### 3.3.1 Host-Driven offloading with DHCPv6

For the NAO host-driven approach, we use Linux-based Nokia N900 mobile handsets as our prototype UEs, running Linux kernel version 2.6.28.10. We enable the IPv6 stack and implement essential components on N900 to support DHCPv6 specific route rules [248] and recursive DNS server selection [253]. Figure 3.15 shows our testing devices. The implementation on the host side includes:

Figure 3.15: Nokia N900 mobile handsets for prototype development.

- Integration and modification of the Internet Systems Consortium (ICS) [2] `DHCPv6 4.2.0` client to support unicast DHCPv6 messages for requesting the DHCPv6 route options [248] and recursive DNS server selection option as specified in Internet Draft *draft-savolainen-mif-dns-server-selection-04* (predecessor of [253]).
- We replace the native DNS resolver (`dnsmasq`) on N900 with the more extensible ICS's `BIND 9.7.1-P2` DNS resolver [3], for which all required behaviors are available through configuration file (`named.conf`) modifications.
- We implement a `DNS Server Selection Configuration Generator` to create `named.conf` based on the DHCPv6 recursive DNS server option.
- We implement an `IP Route Configuration Generator` to install IPv6 routes based on the DHCPv6 route options.
- We develop a set of scripts to place N900 into multi-interfaced state with both cellular and WiFi simultaneously enabled, as by default only one interface is active at a time. The scripts also trigger the

[2] Internet Systems Consortium: `https://www.isc.org/`
[3] BIND DNS software: `https://www.isc.org/downloads/BIND/`

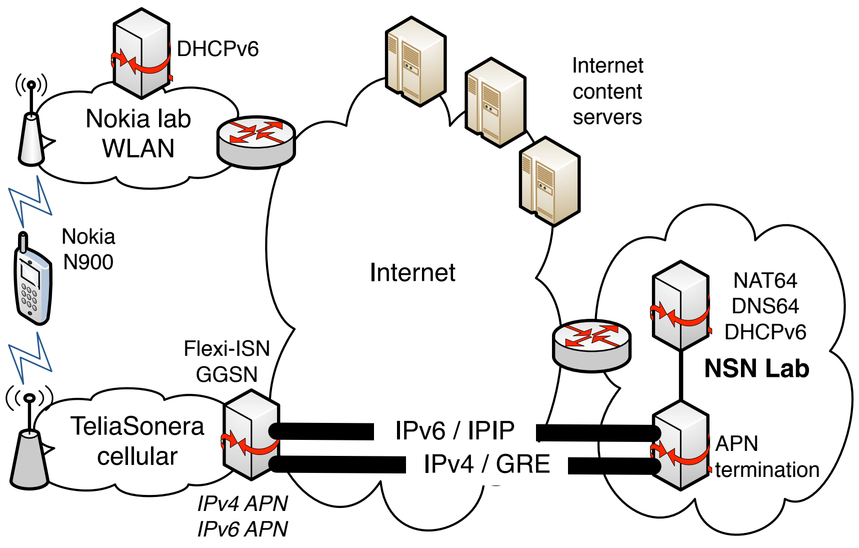


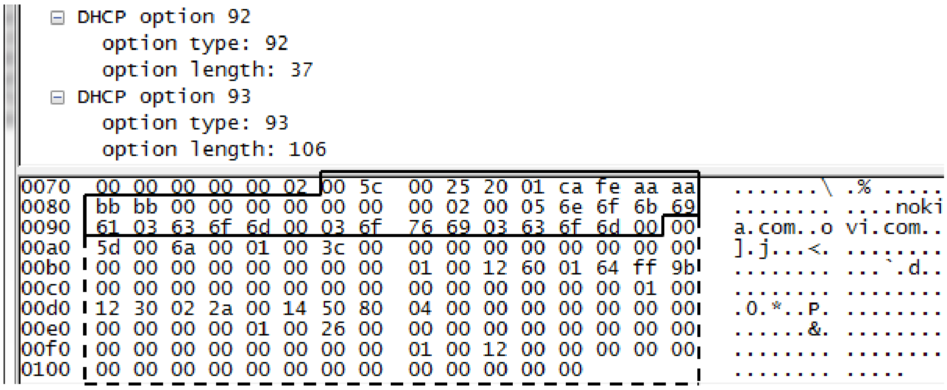
Figure 3.16: Testbed with cellular and WiFi connectivity [26].

activation of BIND, DHCPv6 client, DNS Server Selection Configuration Generator and IP Route Configuration Generator.

To verify our proposal, we set up a testbed in an industry laboratory. As shown in Figure 3.16, dual-stack Internet connectivity via WiFi and cellular accesses are provided to allow us to test dynamic IPv6 traffic offloading. In the testbed, ISC DHCPv6 servers are installed on both access networks. The DHCPv6 server on the cellular access is modified to accept unicast DHCPv6 messages because multicast packet delivery is not available due to the limitation in network setup with tunneling between GGSN and Access Point Name (APN) termination server. In commercial deployment, the DHCPv6 server would reside at the GGSN and hence multicast would be fully supported. For communication, using a secure and trusted channel and/or DNSSEC is recommended to counter possible attackers [253]. We hence utilize the cellular access as a secure and trusted communication channel for conveying offloading information.

As a proof of concept, we conducted live tests for the host-driven approach in the industry laboratory. Figure 3.17 illustrates a Wireshark[4] packet capture of a DHCPv6 Information Reply message with the new options from the operational cellular network. Option codes 92 and 93

[4] Wireshark: `https://www.wireshark.org/`



```
⊟ DHCP option 92
    option type: 92
    option length: 37
⊟ DHCP option 93
    option type: 93
    option length: 106

0070  00 00 00 00 00 02 00 5c  00 25 20 01 ca fe aa aa   .......\ .%.....
0080  bb bb 00 00 00 00 00 00  00 02 00 05 6e 6f 6b 69   ........ ....noki
0090  61 03 63 6f 6d 00 03 6f  76 69 03 63 6f 6d 00 00   a.com..o vi.com..
00a0  5d 00 6a 00 01 00 3c 00  00 00 00 00 00 00 00 00   ].j...<. ........
00b0  00 00 00 00 00 00 00 00  01 00 12 60 01 64 ff 9b   ........ ...`.d..
00c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 01 00   ........ ........
00d0  12 30 02 2a 00 14 50 80  04 00 00 00 00 00 00 00   .0.*..P. ........
00e0  00 00 00 00 01 00 26 00  00 00 00 00 00 00 00 00   ......&. ........
00f0  00 00 00 00 00 00 00 00  01 00 12 00 00 00 00 00   ........ ........
0100  00 00 00 00 00 00 00 00  00 00 00 00 00            ........ .....
```

Figure 3.17: Wireshark capture of recursive DNS server selection (solid line) and specific route (dotted line) DHCPv6 options [26].

were selected for prototyping purposes, as official codes were not available during the experiment period. We also made a live demonstration during Internet Engineering Task Force (IETF) meeting 79 in 2010 at Beijing to show IPv6 traffic offloading between WiFi and a live mobile network even while roaming abroad. In the demonstration, an NAO-enabled N900 was able to use the offloading information fetched from our DHCPv6 server in Finland to configure its DNS and routing table, and then offloaded its IPv6 traffic to the IETF WiFi network. In brief, implementing the NAO host-driven approach on our testing devices is straightforward. The hands-on experience shows that our proposal is feasible.

### 3.3.2 Network-Driven offloading with Neighbor Discovery

By utilizing an IETF protocol [76], the NAO network-driven approach can be deployed on UEs with multiple network interfaces. Several mainstream operating systems have already implemented the extensions, including Linux, BSD variants and Microsoft Windows starting from XP. The network interfaces that accept the new extensions can be specified just using host side configuration. The only consideration is to limit the number of interfaces accepting offloading commands to one interface that is both trusted and centrally managed by the operator. In our case, the 3GPP cellular connection fulfills these requirements.

Figure 3.18 shows the test setup in the industry laboratory dedicated for our experiments. Since modifying a live network GGSN/P-GW is not an option, we implement required tools in the APN termination router as shown in Figure 3.18. When the APN is terminated to an external

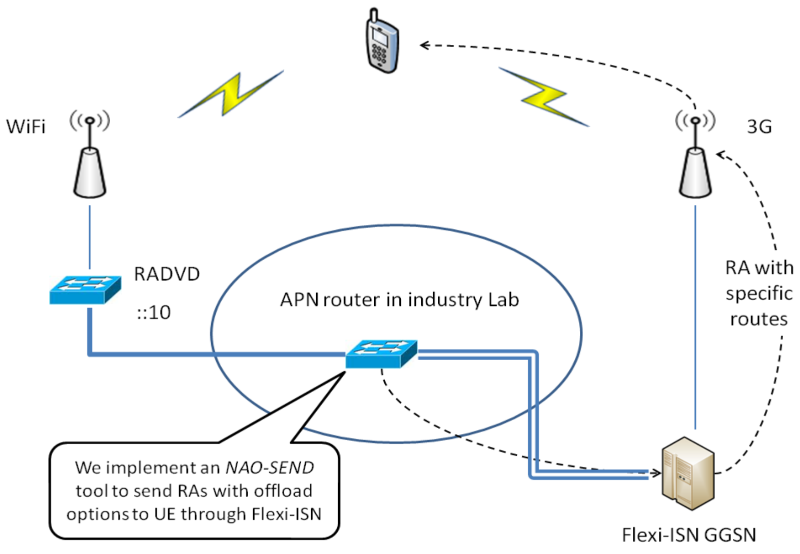
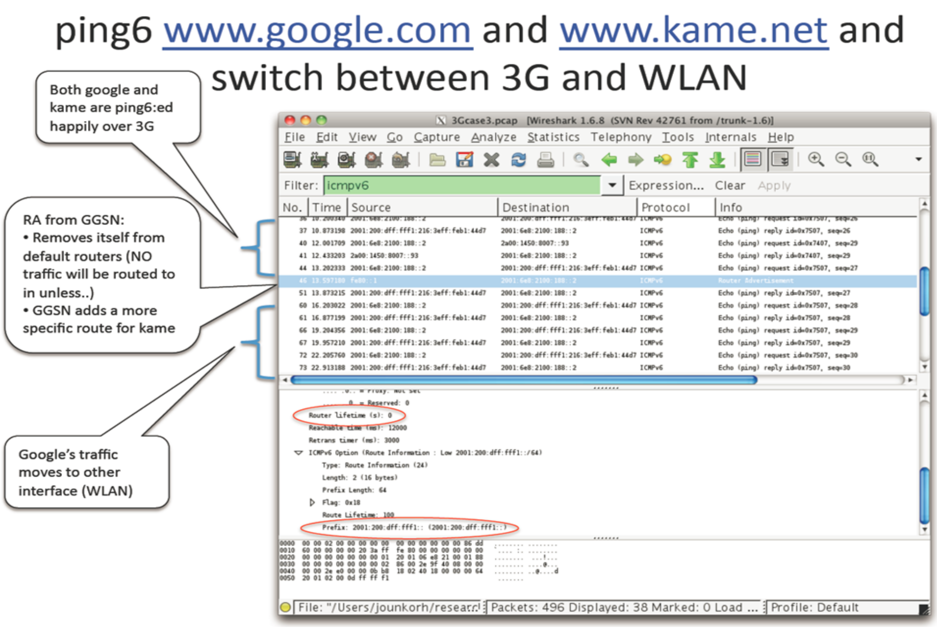

WiFi
3G
RADVD
::10
APN router in industry Lab
RA with specific routes
We implement an *NAO-SEND* tool to send RAs with offload options to UE through Flexi-ISN
Flexi-ISN GGSN

Figure 3.18: Test setup in industry laboratory.

router, the GGSN/P-GW essentially becomes a bridge and the APN terminating router becomes the first-hop router for UEs. We implement an `NAO-SEND` tool that enables us to send RAs with offloading options among other Neighbor Discovery messages, from the cellular network to a specific UE.

Regarding the experiments, host operating systems typically prefer WiFi over cellular access. For instance, Linux-based systems implicitly prefer WiFi access over cellular access, thus prioritizing the first-hop router(s) in WiFi access over the first-hop router in a cellular access. For consistency, we always use *LOW* default router priority for the cellular access. Because the default router priority on other interfaces is implicitly *MEDIUM*, the host IP stack will prefer any other interface for default destinations rather than cellular access. When a cellular operator wants to route certain traffic over the cellular, it only needs to send an RA with an RIO containing the IPv6 prefix(es) for those destinations. The default router and default address selection algorithm [244] in the host IP stack will take care of selecting an appropriate interface for new IPv6 connections (e.g., existing TCP connections will not be moved). By adopting the model of operation '*offload everything except specific destinations*', we also avoid a large number of routing rules installed on UEs as overhead.

To test network-driven IPv6 traffic offloading, we conducted experi-


ping6 www.google.com and www.kame.net and switch between 3G and WLAN
Both google and kame are ping6:ed happily over 3G
RA from GGSN: • Removes itself from default routers (NO traffic will be routed to in unless..) • GGSN adds a more specific route for kame
Google's traffic moves to other interface (WLAN)
3Gcase3.pcap [Wireshark 1.6.8 (SVN Rev 42761 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: icmpv6
Expression... Clear Apply

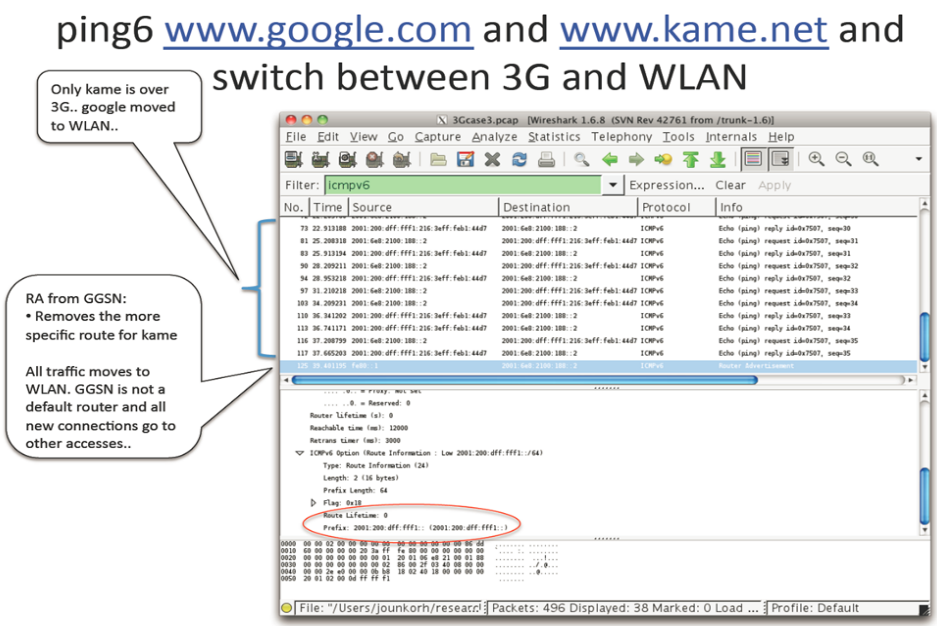
```
No. Time      Source                                 Destination                            Protocol Info
 37 10.873198 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=26
 40 12.001709 2001:6e8:2100:188::2                   2a00:1450:8007::93                     ICMPv6   Echo (ping) request id=0x7407, seq=29
 41 12.433203 2a00:1450:8007::93                     2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7407, seq=29
 44 13.202333 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=27
 46 13.597180 fe80::1                                2001:6e8:2100:188::2                   ICMPv6   Router Advertisement
 51 13.873215 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=27
 60 16.203022 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=28
 61 16.877199 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=28
 66 19.204356 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=29
 67 19.957210 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=29
 72 22.205760 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=30
 73 22.913188 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=30

        .... ...0 = Reserved: 0
      Router lifetime (s): 0
      Reachable time (ms): 12000
      Retrans timer (ms): 3000
    ▽ ICMPv6 Option (Route Information : Low 2001:200:dff:fff1::/64)
        Type: Route Information (24)
        Length: 2 (16 bytes)
        Prefix Length: 64
      ▷ Flag: 0x18
        Route Lifetime: 100
        Prefix: 2001:200:dff:fff1:: (2001:200:dff:fff1::)

0000  00 00 02 00 00 00 00 00  00 00 00 00 00 00 86 dd
0010  60 00 00 00 00 20 3a ff  fe 80 00 00 00 00 00 00
0020  00 00 00 00 00 00 00 01  20 01 06 e8 21 00 01 88
0030  00 00 00 00 00 00 00 02  86 00 2e 9f 40 08 00 00
0040  00 00 2e e0 00 00 0b b8  18 02 40 18 00 00 00 64
0050  20 01 02 00 0d ff ff f1

File: "/Users/jounkorh/researc | Packets: 496 Displayed: 38 Marked: 0 Load ... | Profile: Default
```

Figure 3.19: Using RIO to remove a default router and add a specific route.


ping6 www.google.com and www.kame.net and switch between 3G and WLAN
Only kame is over 3G.. google moved to WLAN..
RA from GGSN: • Removes the more specific route for kame
All traffic moves to WLAN. GGSN is not a default router and all new connections go to other accesses..
3Gcase3.pcap [Wireshark 1.6.8 (SVN Rev 42761 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: icmpv6
Expression... Clear Apply

```
No.  Time      Source                                 Destination                            Protocol Info
 73  22.913188 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=30
 81  25.208318 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=31
 83  25.913194 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=31
 90  28.209211 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=32
 94  28.953218 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=32
 97  31.210218 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=33
103  34.209231 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=34
110  36.341202 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=33
113  36.741171 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=34
116  37.208799 2001:6e8:2100:188::2                   2001:200:dff:fff1:216:3eff:feb1:44d7   ICMPv6   Echo (ping) request id=0x7507, seq=35
117  37.665203 2001:200:dff:fff1:216:3eff:feb1:44d7   2001:6e8:2100:188::2                   ICMPv6   Echo (ping) reply id=0x7507, seq=35
125  39.401195 fe80::1                                2001:6e8:2100:188::2                   ICMPv6   Router Advertisement

        .... ..0. = Reserved: 0
      Router lifetime (s): 0
      Reachable time (ms): 12000
      Retrans timer (ms): 3000
    ▽ ICMPv6 Option (Route Information : Low 2001:200:dff:fff1::/64)
        Type: Route Information (24)
        Length: 2 (16 bytes)
        Prefix Length: 64
      ▷ Flag: 0x18
        Route Lifetime: 0
        Prefix: 2001:200:dff:fff1:: (2001:200:dff:fff1::)

0000  00 00 02 00 00 00 00 00  00 00 00 00 00 00 86 dd
0010  60 00 00 00 00 20 3a ff  fe 80 00 00 00 00 00 00
0020  00 00 00 00 00 00 00 01  20 01 06 e8 21 00 01 88
0030  00 00 00 00 00 00 00 02  86 00 2f 03 40 08 00 00
0040  00 00 2e e0 00 00 0b b8  18 02 40 18 00 00 00 00
0050  20 01 02 00 0d ff ff f1

File: "/Users/jounkorh/researc | Packets: 496 Displayed: 38 Marked: 0 Load ... | Profile: Default
```

Figure 3.20: Using RIO to remove a specific route.

ments in the laboratory. We used the `NAO-SEND` installed on the APN router to send RAs to a testing UE through the Flexi-ISN GGSN located in the operator network. As a proof of concept, the testing traffic consists of ICMPv6 ping packets sent from the UE to 'www.google.com' (2a00:1450:8007::93) and 'www.kame.net' (2001:200:dff:fff1:216:3eff:feb1:44d7). Regarding the testing UE, we enabled IPv6 on its WiFi and cellular interfaces. Both the first-hop routers in the WiFi and cellular networks were added to the UE's default router list. For testing purpose, we further set the default router preference of the WiFi router as MEDIUM and the preference of the cellular router as HIGH.

Figure 3.19 and 3.20 show the Wireshark packet capture on the UE. As illustrated in Figure 3.19, all the ping packets to 'www.google.com' and 'www.kame.net' were initially sent through the cellular (3G) network. We first instructed the `NAO-SEND` to send an RA for offloading, which contained an RIO with a specific route for 'www.kame.net' (2001:200:dff:fff1::). Since we deliberately set the 'Router Lifetime' in this RA to 0, once the UE received the RA, it removed the cellular first-hop router from its default router list. In addition, our NAO-enabled UE was able to parse the RIO and add a specific route (2001:200:dff:fff1::) for the cellular router to its routing table. Since the cellular default router was removed and there was not specific route for 'www.google.com', the ping packets to 'www.google.com' were offloaded to the WiFi network. The ping packets to 'www.kame.net' were stilled sent through the cellular network according to the specific route suggested by the RIO. As shown in Figure 3.20, we sent another RA to remove the specific route of 'www.kame.net' by setting the 'Router Lifetime' of 2001:200:dff:fff1:: as 0. After UE removed this specific route of cellular network, the ping packets to 'www.kame.net' were also offloaded to WiFi network.

In brief, the live experiments show that we can use the proposed approach to support IPv6 traffic offloading. We note that 3GPP architecture relies on IPv6 Stateless Address Auto Configuration (SLAAC) for its (un)trusted 3GPP access. The GGSN/P-GW must always send Router Advertisements for SLAAC purposes and therefore using the 3GPP access as a command channel for offloading purposes is an incremental enhancement to the GGSN/P-GW functionality. Furthermore, RAs can always be sent unsolicited. According to our experiments, the NAO solution allows for an on-demand push mode of operation from the cellular operator point of view.

### 3.3.3 Network-Driven offloading with IPv4 support

Based on our design principle to minimize the impact on the host system [23], we implement a system prototype for IPv4 traffic offloading with Linux kernel version 3.0. The system architecture is illustrated in Figure 3.21. On the host side, we extend the existing kernel implementation of Neighbor Discovery protocol (*ndisc.c*) by adding an intercepting hook function (*kernel offload hook*) and *kernel offload module* to push more-specific IPv4 routes and IPv4 default gateway information contained in RA options from the kernel to user space via the *sysfs interface*. In the user space, we develop an *IPv4 offload daemon* that handles main tasks for IPv4 traffic offloading by manipulating IPv4 routing tables.

On the network side, we implement the NAO protocol specified in Internet-Draft draft-korhonen-mif-ra-offload-03 [24] (the predecessor of [25]) and integrate it into the tool `NAO-SEND`. For the protocol implementation, we use the RIO Prefix field [76] to convey the specific route through an IPv4-mapped IPv6 address. We install `NAO-SEND` on an APN server which is connected to the Flexi-ISN GGSN located in the operator network.

As presented in Figure 3.21, the deployment of the NAO framework depends on the collaboration of network and host side. On receiving an RA with IPv4 offload options, the NAO *kernel offload hook* will parse the message and notify the *kernel offload module*. Based on the offload options, the *kernel offload module* will set up the *sysfs interface* and record the offloading information. Once the recording is completed, the *kernel offload module* will notify *IPv4 offload daemon* with a pointer to the *sysfs interface*. The NAO *IPv4 offload daemon* in the user space checks the input in the *sysfs interface* and modifies routing tables according to the offloading policies.

To test the network-driven IPv4 traffic offloading, we set up a test environment in our laboratory at the University of Helsinki. Figure 3.22 shows the test setup. The network layout for this experiment is similar to the one shown in Figure 3.16. We installed our kernel extensions on the test machine, which is connected to cellular access through N900. The WiFi access to the Internet is provided by a laptop machine. We use the `NAO-SEND` tool to send RAs from the APN router to UEs through the Flexi-ISN GGSN located in the operator network.

As proof of concept, we conducted ping tests similar to the procedure used for RA-based IPv6 offloading. Figure 3.23 shows a captured RA message, where the RIO option carries an IPv4-mapped IPv6 address (`::ffff:203.178.141.0/120`) to route an IPv4 subnet to the RA originating router. The RA also carries the default IPv4 gateway address of the

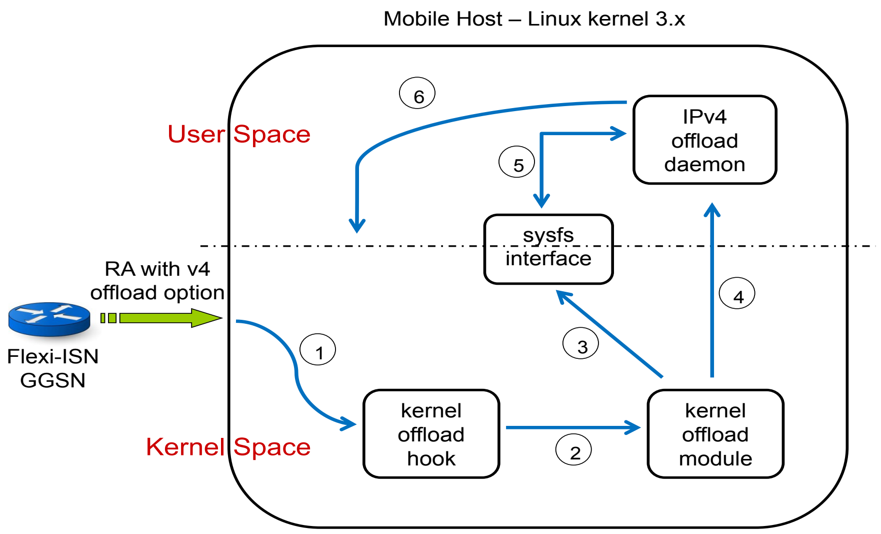


Figure 3.21: NAO Implementation to support IPv4 traffic offloading [26].

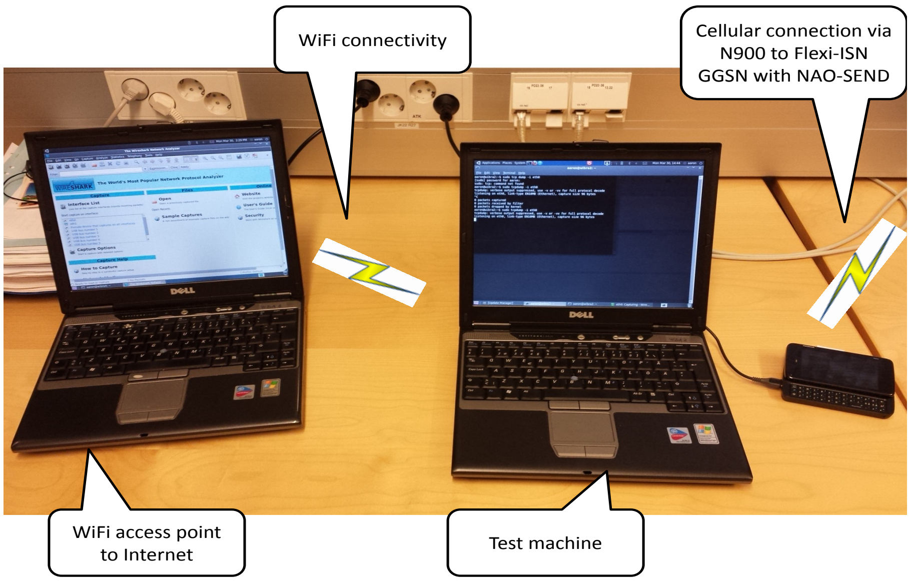
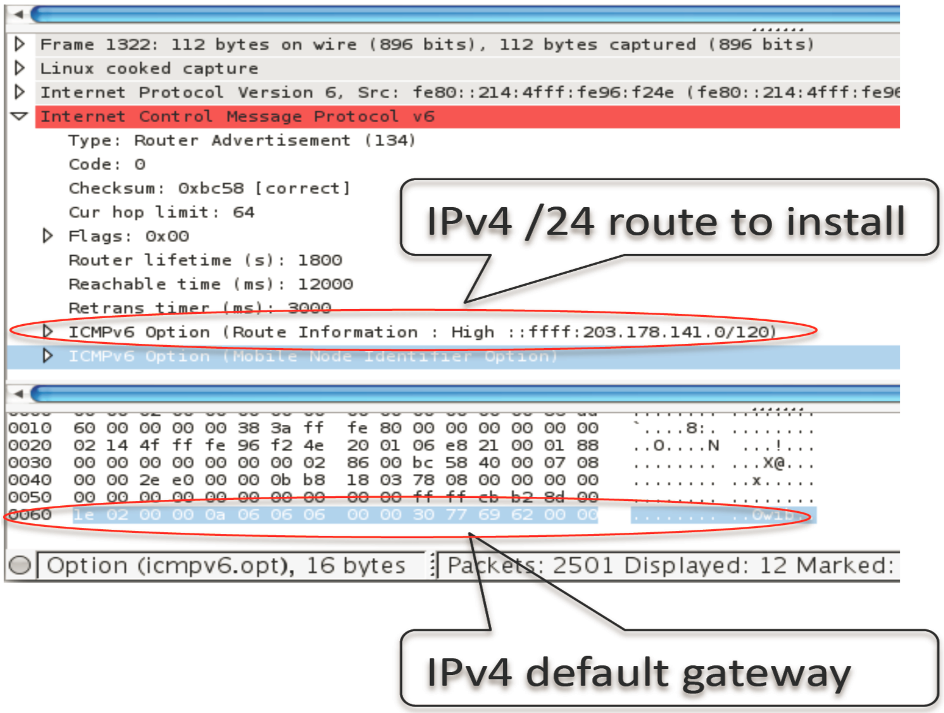

WiFi connectivity
Cellular connection via N900 to Flexi-ISN GGSN with NAO-SEND
WiFi access point to Internet
Test machine

Figure 3.22: Test setup for NAO IPv4 offloading.



```
▷ Frame 1322: 112 bytes on wire (896 bits), 112 bytes captured (896 bits)
▷ Linux cooked capture
▷ Internet Protocol Version 6, Src: fe80::214:4fff:fe96:f24e (fe80::214:4fff:fe96
▽ Internet Control Message Protocol v6
     Type: Router Advertisement (134)
     Code: 0
     Checksum: 0xbc58 [correct]
     Cur hop limit: 64
   ▷ Flags: 0x00
     Router lifetime (s): 1800
     Reachable time (ms): 12000
     Retrans timer (ms): 3000
   ▷ ICMPv6 Option (Route Information : High ::ffff:203.178.141.0/120)
   ▷ ICMPv6 Option (Mobile Node Identifier Option)

0010  60 00 00 00 00 38 3a ff  fe 80 00 00 00 00 00 00   `....8:. ........
0020  02 14 4f ff fe 96 f2 4e  20 01 06 e8 21 00 01 88   ..O....N ....!...
0030  00 00 00 00 00 00 00 02  86 00 bc 58 40 00 07 08   ........ ...X@...
0040  00 00 2e e0 00 00 0b b8  18 03 78 08 00 00 00 00   ........ ..x.....
0050  00 00 00 00 00 00 00 00  00 00 ff ff cb b2 8d 00   ........ ........
0060  1e 02 00 00 0a 06 06 06  00 00 30 77 69 62 00 00   ........ ..0wib..

Option (icmpv6.opt), 16 bytes   Packets: 2501 Displayed: 12 Marked:
```

Figure 3.23: Packet capture showing an RA with RIO carrying an IPv4-mapped IPv6 address and an IPv4 default gateway address.

dual-stacked router as the last option (`10.6.6.6` i.e. `0a 06 06 06`).

To sum up, our experiments in operational networks show that a cellular operator can take advantage of the cellular network connection as a secure command channel to push offloading policies into the UE for both IPv6 and IPv4 traffic, while still only using standard IETF protocols.

## 3.4 Solution Comparison and Discussion

Table 3.2 summarizes the key characteristics of the 3GPP-based solutions discussed in Section 2.2.2 and our traffic offloading solutions in NAO framework [26].

- '*IPv4/IPv6*' indicates the IP version a proposal applies to.
- '*Dynamic*' characterizes the nature of offloading and policy information dynamics, which indicates whether such policies can be easily updated during an ongoing data session.

Table 3.2: Solution comparison [26].

| Proposal | IPv4/v6 | Dynamic | Push/Pull | UE | Core | Offload |
|---|---|---|---|---|---|---|
| MAPCON [111] | Both | No | Push | Yes | Yes | R |
| SIPTO [113] | Both | No | Push | Yes | Yes | C |
| LIPA [113] | Both | No | Push | Yes | Yes | R,C |
| IFOM [114] | Both | Yes (DSMIP) | Pull/Push | Yes | Yes | R |
| SaMOG [116] | Both | No | Push | Yes | Yes | R |
| S2b [58] | Both | Yes (IKEv2) | Push | Yes | Yes | R |
| NAO DHCPv6 | IPv6 | UE init | Pull | No | No | R,C |
| NAO RFC4191 | IPv6 | Yes | Push | No | IP | R,C |
| NAO IPv4 | IPv4 | Yes | Push | No | IP | R,C |

- '*Push/Pull*' indicates whether the network pushes the offloading policies down to the UEs or the UEs pull network-assisted information from the network.

- '*UE*' indicates whether the solution involves UE functions that are specific to 3GPP architecture.

- '*Core*' indicates whether 3GPP packet core nodes and 3GPP protocols are impacted or not. 'IP' means changes on IP stack.

- '*Offload*' shows what the offloading targets are. 'R' means radio access and 'C' means core network.

As of today, the use of ANDSF [58] cannot be considered a dynamic interface for frequent policy updates on ongoing data sessions. Full dynamic support would require a deployment of the optional 3GPP Generic Bootstrapping Architecture (GBA) and a short message service (SMS) on top of ANDSF in both network and UE for push operation. Therefore, all proposals that are dependent on the ANDSF or OMA Device Manager (DM) function are not considered dynamic, including MAPCON, SIPTO, LIPA, and SaMOG. Meanwhile, IFOM and S2b are considered dynamic as they adopt IP layer mechanisms, i.e., Dual Stack Mobile IP (DSMIP) [71, 72] and IKEv2 [254], respectively. All three NAO solutions are dynamic for the flexible Internet layer design with network assistance.

3GPP-based proposals are 'Push' oriented. In case of IFOM the traffic policies are part of the DSMIPv6 protocol signaling, and policies are mainly requested by the UE. It is hence considered 'Pull'. However, if the ANDSF or DM is present, then policies can also be pushed from the network. By using DHCPv6 and Neighbor Discovery, the NAO framework supports both 'Push' and 'Pull'.

Regarding the '*UE*' aspect that indicates whether the solution involves UE functions that are specific to 3GPP architecture, all 3GPP solutions are considered to be positive. The same applies to '*Core*' regarding the listed solutions in the table. Meanwhile, the NAO solutions are independent of the access technology. The NAO's IPv6 (RFC4191) and IPv4 approaches involve IP stack changes in the network (marked with 'IP' in the table).

For the '*Offload*' target, SIPTO is mainly for core networks, while MAPCON, IFOM, SaMOG, and S2b target radio accesses. LIPA supports both core network and radio access. The NAO solutions can be applied to offload traffic for both core networks and radio accesses.

Regarding the NAO DHCPv6 approach, the DHCPv6 client in the UE has to initiate a DHCPv6 protocol exchange in order for the network to push new routes and policies to the UE. However, if the network-side reconfiguration feature is enabled, then the network can trigger the UE to perform a generic reconfiguration of its IP configuration. Since both ends have to agree on the reconfiguration support, we consider the DHCPv6 solution a host-driven, i.e., 'Pull' mode. If a GGSN/P-GW implements a DHCPv6 relay, then there is no impact to the 3GPP packet core.

Both NAO IPv6 and IPv4 approaches build on top of the Neighbor Discovery protocol to push offloading policies through RAs. The RFC4191-based solutions are lightweight and simple to implement, specifically when the network-side policies are taken into account. The RFC4191 support can already be found in many mainstream OSes. Specifically, it is a good match in 3GPP networks, because SLAAC is the only mandatory IPv6 configuration method for 3GPP accesses. The real downside for RFC4191-based solutions is that it is difficult for the network side to know whether UEs support the feature or not. The DHCPv6-based approach can notify the network side regarding the supported feature set.

Within the NAO framework, the DHCPv6-based solution lacks a proper 'Push' feature for the dynamic policy updates and it is not a mandatory function in 3GPP architecture. The RFC4191-based solutions integrate easily into 3GPP architecture and the network-side policy push is easy to implement and especially lightweight. However, similar to the case of DHCPv6, the original RFC4191-based solution has no support for IPv4 traffic. Extending RFC4191 with IPv4 knowledge is easy on the network side but it requires UE-side changes and successful standardization in order to get mainstream OS vendors to support it. From the protocol perspective, solutions in NAO try to conform to a 'pure IP' view. None of the solutions aim at guaranteeing that the offloading policy provided by the network would work in all possible cases. More important is that the UE always has Internet connectivity, meaning none of the access networks used shall be a walled garden.

Compared with the research-oriented proposals discussed in Section 2.2.2, our solutions do not break backward compatibility, work in the majority of the intended use cases, and would not require mobile devices to implement technology-specific extensions such as restricted to 3GPP architecture. Compared with 3GPP solutions, the implementation experience and the comparison shows that the solutions using IETF-only technologies are feasible and lightweight.

By using network assistance, our proposal offers several advantages. First, the assistance from the network side in terms of user location, WiFi AP location, and user mobility history is valuable to enhance the offloading decision and efficiency. Second, energy efficiency can be improved by shifting offloading preparation to the network side. Third, by following the IETF standard track development, the deployability of our proposal will suffer less incompatibility issues owing to its openness.

The advance of smartphones and tablets also strengthens our IP offloading visions, since even most mid-range UE devices have a WLAN radio. There are a few issues left, though. First, UE may not allow operating multiple radios simultaneously, which effectively prohibits selective offloading of IP traffic between access technologies. In NAO, we assume UE can operate multiple radios in parallel. The second issue is how to determine which IP traffic to offload and which traffic to route through the mobile operator's core. Third is how to steer the offloading decision making from the network side. This can be challenging, especially in the case of split-UE devices. The fourth issue is how to minimize the impact on the operator network and especially in the UE.

Regarding the impact of bulk TCP traffic on delay-sensitive traffic (discussed in Section 3.1), NAO can assist operators to offload traffic sources that are identified as bulk data transfer. Although how to distinguish and offload bulk TCP traffic in a fine-grained manner is still open, NAO provides feasible mechanisms to alleviate the impact of competing traffic in mobile access by selectively offloading traffic sources generating bulk data transfer to other alternative networks. We note that NAO is an extensible framework and research proposals [5, 7, 8, 10, 11, 85, 86, 87] can be integrated on either network side or UEs to optimize the performance of mobility prediction and energy saving. In particular, such collaboration will greatly help in the case of open WiFi accesses where the history-based prediction independent of the prior knowledge of WiFi AP locations can compensate and extend the offloading to open access areas. Because NAO is not restricted to WiFi only, the support of opportunistic communications can be included to the framework to further promote its flexibility in traffic offloading.

## 3.5 Summary

In NAO we aim at an intelligent and lightweight traffic management, especially how to bypass the mobile operator core network and offload IP traffic efficiently from the mobile operator network to alternate cheaper accesses as soon as possible. The proposed solutions focus on the Internet layer approach suitable for telecommunication systems.

The design of NAO is driven by our measurement studies in operator networks. The observations deepen our understanding of the key issues and offer insights to guide the traffic offloading design. We test our solutions in operator environments and show its practical feasibility through live experiments. We have also compared our DHCPv6 and IPv6 Neighbor Discovery based solutions against 3GPP standardized offloading solutions. Our framework brings up the possibility and reveal the potential of what network assistance can do for offloading mobile traffic. The implementation experience and the comparison show that the solutions using IETF-only technologies are feasible and lightweight to deploy both on the network side and on mobile devices.

We believe that a deployable IP traffic offloading solution will be a mixture of existing technologies standardized in 3GPP and IETF, and what modern IP stacks can do. It is unlikely that mainstream operating systems' IP stack would implement 3GPP-specific technologies. Different third party dialer software may then add those missing elements.

Being our first step to enable collaborative traffic offloading, we verify the feasibility of NAO by running live experiments in the operator networks. The observations and lessons from this work provide insights for follow-up work that explored how to enable collaborative energy awareness in traffic offloading.

# Chapter 4

# Energy-Aware Traffic Offloading

This chapter presents the architecture of Metropolitan Advanced Delivery Network (MADNet) [16] that aims at improving the energy efficiency of traffic offloading for ***smartphones***. First, Section 4.1 describes the motivation and highlights the importance of energy awareness in traffic offloading. Section 4.2 presents our measurement studies that explore the energy cost and performance in WiFi-based traffic offloading. These observations provide valuable insights for the design of MADNet energy-aware traffic offloading. Section 4.3 presents the system requirements and design. Section 4.4 illustrates our implementation and experiments in live networks. Section 4.5 discusses related work and key issues in energy-aware traffic offloading. Finally, Section 4.6 summarizes the chapter.

## 4.1 Motivation

As discussed in Chapter 3, mobile traffic offloading is regarded as a promising approach to relieve this problem by using complementary communication technologies such as WiFi to offload the traffic originally planned for transmission over cellular networks. Although WiFi-based offloading has been shown to be effective [5, 6], there are still several issues remaining to address when offloading mobile traffic for smartphones. First, existing proposals such as Wiffler [5] and oSCTP [8] are designed mainly for PC-alike laptops on vehicular networks without considering the energy consumption of offloading. Lee et al. [6] evaluated the energy saving of offloading through trace-driven simulations with several simplified assumptions, but how to harvest the energy gain of mobile traffic offloading ***in practice*** is still an open problem. Second, through our extensive war-driving and war-walking measurements using smartphones in four cities of US and Europe, we found that the number of open-accessible WiFi access points (APs) was very limited, usually less than 2%. Therefore the offloading schemes that use only open APs may not be enough. Finally, previous work focused on increasing the amount of delay-tolerant data traffic that can be offloaded to WiFi networks. However, delay-tolerant applications generate only a small amount of data traffic compared with streaming applications [126]. Moreover, mobile users may have different delay-tolerant thresholds for different applications [166].

To address these challenges, we propose MADNet energy-aware offloading, a novel traffic offloading architecture for smartphones. The key goal of our proposal is to extend the smartphone battery life in traffic offloading. According to our measurements (discussed in Section 4.2), transferring the same amount of data may consume more energy through WiFi access with weak signal strength than transferring the data over a good quality 3G access. If a scheme merely aims at increasing the amount of offloaded traffic to WiFi networks without considering the energy consumption on smartphones, it can drain the battery much faster, which significantly degrades the user experience. Furthermore, due to the limitation of WiFi antennas deployed on existing smartphones, offloading mobile traffic for smartphones is much more challenging than that offloading traffic for PCs (discussed in Section 4.2). In order to compensate the existing restrictions, we need a dedicated scheme.

The above observations motivate the design of the MADNet energy-aware offloading. MADNet harvests the collaboration across cellular operators, WiFi providers, and end users to improve the energy awareness of offloading for smartphones. In our work, we aim to explore the following aspects:

- We need to investigate and compare the performance of WiFi Internet access in metropolitan areas for different types of mobile systems such as netbooks and smartphones. The observations can offer valuable insights for future design and inspire us to exploit new solutions, such as utilizing the predictable nature of streaming content to enable data prefetching, which can alleviate the negative impact of hardware limitations on smartphones.

- We need an efficient and deployable *energy-aware* algorithm to assist the offloading decision. The design of the algorithm can reflect the insights obtained from our measurements in metropolitan WiFi accesses. Being a core part of the system, the algorithm should be able to tolerate minor errors of input parameters such as user location, mobility prediction, and estimation of system offloading capacity.

- Based on the design, a prototype implementation can show the feasibility of our proposal and shed light on how to deploy it in existing environments. We further conduct field experiments to verify our proposal and show the efficacy of our system in terms of performance and energy saving.

We advocate that MADNet can bring us three advantages. First, the collaborative design brings together WiFi and cellular networks in a coordinated manner which can benefit traffic offloading through new mechanisms that harvest the collaboration. For instance, coordinated data prefetching at WiFi APs may improve the utilization of WiFi channels and thus reduce the energy consumption on smartphones, as the capacity of WiFi links is usually higher than the available capacity across the Internet [201] and WiFi is known to be more energy efficient than 3G [130]. Second, the participation of WiFi providers and end users can significantly increase the offloading opportunities by bringing more accessible WiFi APs and using opportunistic communications [93]. Finally, by combining the above two factors, we can also increase the amount of mobile traffic that can be offloaded from cellular to WiFi.

## 4.2 Measurement Study

Because we focus on WiFi-based traffic offloading, we aim to answer the following three questions through our measurements:

- *Is it feasible to offload mobile data traffic to WiFi networks, given the current deployment and availability of WiFi APs in metropolitan areas?*
- *Is the performance of WiFi-based Internet access for moving smartphones good enough to offload mobile traffic?*
- *What is the impact of traffic offloading for smartphones in terms of energy consumption?*

### 4.2.1 Metropolitan WiFi access and energy consumption

In this study, we explore three domains: 1) Accessible WiFi APs in metropolitan areas. 2) WiFi-based Internet access for moving smartphones. 3) Energy cost in traffic offloading.

Table 4.1: Statistics of detected APs in Berlin, Chicago and Baltimore.

| City | Berlin | Chicago | Baltimore |
|---|---|---|---|
| Speed | driving | walking | driving |
| Detected APs | 4421 | 4588 | 3418 |
| APs without encryption | 351 (7.9%) | 775 (16.9%) | 621 (18.2%) |
| APs granting IP addresses | 12 (0.27%) | 18 (0.39%) | 6 (0.18%) |
| Accessible APs for offloading | **0** (**0.0%**) | **7** (**0.15%**) | **1** (**0.03%**) |

### *Accessible WiFi APs in Metropolitan Areas*

We conducted field studies in three major cities, Berlin in Germany (August 2010), Chicago (September 2010) and Baltimore (November 2010) in the US, using a tool called `3G-WiFi`. The measurement tool consists of two threads: the first thread measures TCP uplink and downlink throughput of the access network by communicating to a local reference server; the second thread scans neighboring WiFi APs and then calls the first thread to test the access networks for all open-accessible APs. `3G-WiFi` conducts tests over access networks periodically with 10-second intervals. We installed `3G-WiFi` on Nokia N900 smartphones for this measurement study. The default operating system of N900 is Maemo 5, an open-source Linux distribution (2.6.28 kernel). The WiFi chipset used by N900 is Texas Instruments WL1251.

We chose four areas in Berlin for the war-driving tests, including two neighborhoods near Nokia Siemens Networks and Schloss Charlottenburg, along Kurfürstendamm avenue and along Unter den Linden boulevard (two of the most popular avenues in Berlin). In Chicago, the war-walking test was around Michigan Avenue. We also conducted war-driving in the downtown area of Baltimore. We summarize the statistics of detected APs in these three cities in Table 4.1. As presented in the table, *Detected APs* reflect the total number of WiFi access points that were detected through WiFi scanning. *APs without encryption* are the 'open' WiFi access points that we can associate without using WiFi Protect Access (WPA), WiFi Protect Access II (WPA2), or Wired Equivalent Privacy (WEP) [255, 256]. *APs granting IP addresses* are the APs that granted IP addresses to the associated mobile devices. *Accessible APs for offloading* are those that allow `3G-WiFi` to connect to our reference server and test at least one of

Table 4.2: Statistics of detected APs in College Park, MD.

| Timeout | 1 s | 2 s | 3 s |
|---|---|---|---|
| Detected APs | 581 | 647 | 519 |
| APs without encryption | 87 (15.0%) | 80 (12.4%) | 59 (11.4%) |
| APs granting IP addresses | 0 (0.0%) | 3 (0.46%) | 1 (0.19%) |
| Accessible APs for offloading | **0** (**0.0%**) | **1** (**0.15%**) | **0** (**0.0%**) |

the metrics, i.e., TCP uplink or downlink throughput, thereby showing the Internet accessibility.

We note that Nicholson et al. [87] have reported that about 40% of APs detected in three neighborhoods of Chicago were open in 2006 and Wiffler [5] also found that around 40% of APs encountered by 20 public transit vehicles in Amherst MA were not encrypted. In contrast with these results, the percentage of APs without encryption (i.e., 'open APs') in our measurement is very low, less than 10% in Berlin and less than 20% in Chicago and Baltimore. One of the possible reasons is that more and more people become aware of the security of their home networks, especially in big cities.[1]

An interesting finding is that most of these 'open' APs are not accessible, as they may apply MAC address filter or web-based authentication for access control. The percentage of *Accessible APs for offloading* is extremely low, less than 1%, which poses a challenge to the offloading schemes that rely only on open-accessible APs.

In order to detect and measure as many APs as possible, we set the time-out of DHCP messages and TCP connection setup to be 0.1 seconds. We are aware that these low values may affect the measurement results [257]. Thus, we also conducted war-driving in a neighborhood of College Park, MD with three different timers (1, 2 and 3 seconds) for DHCP discovery messages and TCP connection setup. The driving speed was ~20–30 km/h. Table 4.2 presents the results, which show that even in a small college town, the percentage of open APs is less than 20%. Moreover, there is almost no open-accessible APs in that neighborhood. Therefore, *active participation from end users (by sharing their home APs) is a key enabling factor for*

[1]We confirmed that in the US most of the home APs distributed by AT&T U-verse and Verizon FiOS are encrypted by default, with their technical support of customer services. The same holds true for Deutsche Telekom T-Home in Germany.

Table 4.3: Identifiable patterns of ESSIDs for detected APs in Berlin.

| ESSID Pattern | Provider | Amount |
|---|---|---|
| tmobile | Deutsche Telekom | 622 |
| ALICE-WLAN | Alice DSL | 215 |
| Arcor | Vodafone | 68 |
| EasyBox | | 117 |
| o2DSL | Telefonica | 35 |

*mobile data offloading*, as end users can also be WiFi providers as part of the MADNet architecture.

Through the identifiable patterns of ESSID (Extended Service Set IDentifier) from the collected trace of AP information in Berlin, we found that there are a large number of APs ($\sim$1000) deployed/distributed by cellular operators. Table 4.3 shows the patterns of ESSIDs for the detected APs in Berlin. We also list the corresponding service providers (inferred through these ESSIDs) of these WiFi APs. As we can see in this table, there are a large number of home APs distributed by cellular operators in Berlin. Moreover, we found 426 APs with ESSID "FRITZ!Box" which are produced by a very popular electronics company in Germany, AVM GmbH [2].

Besides the Berlin trace, we analyzed the traces collected from Chicago and Baltimore US. We identified 616 WiFi APs produced by 2Wire, Inc. and distributed by AT&T to home users from the Chicago trace. From the Baltimore trace, we identified 334 WiFi APs produced by Westell or Abocom and distributed by Verizon.

We note that these numbers show lower bounds because users may change the ESSID after they get the APs from their providers, and the number close to ground-truth can be even higher. This observation indicates that as operators are deploying their WiFi APs, *a collaborative WiFi offloading design can benefit both cellular operators and WiFi providers to share their resources and thereby improve the possibility to offload more traffic away from overloaded cellular access.*

### *WiFi-based Internet Access for Moving Smartphones*

To investigate the performance of WiFi-based Internet access for moving smartphones, we conducted field experiments at both walking and driving speeds in College Park, Maryland, USA. Similar to the setup used by

[2] `http://www.avm.de/en/`

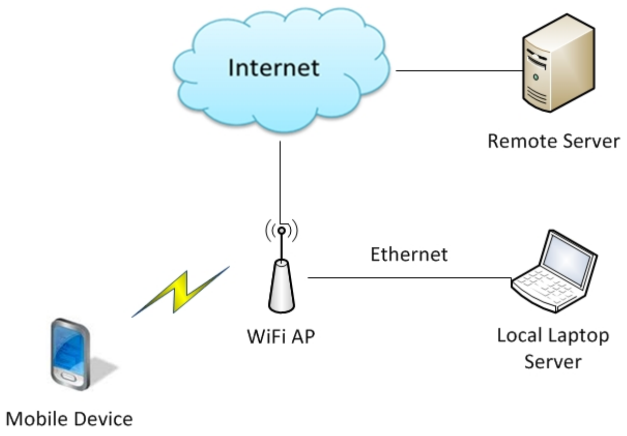


Figure 4.1: Experiment setup.

Hadaller et al. [257], Figure 4.1 illustrates our test setup and equipment. We used two servers (local and remote) for performance measurement. The remote server was located in an industrial lab in Berlin. The local server was running on a laptop. The laptop was connected to a WiFi AP via 100M Ethernet. The AP was also connected to a campus network for Internet access to reach the remote server. Regarding IP address configuration, DHCP was supported by the WiFi AP.

Compared with previous work [102, 257], the major differences are that we used a smartphone (N900) as the client, instead of a laptop, and we experimented with both walking and driving speeds. The experiments were carried out in an apartment neighborhood in College Park, MD. Figure 4.2 shows the map for these experiments.

First, we conducted a group of outdoor experiments for moving smartphones at walking speed, passing by the 'Home AP', which was connected to our local server. The AP was around 36 meters away from the road. To guarantee that the duration of experiment was longer than the actual AP association time of the smartphone, we walked from one location to another along the road for the selected trail (the blue line as shown in Figure 4.2). Both locations were out of the coverage area of the AP. We performed experiments for both directions along the road, to the east and to the west. We measured this scenario against three settings: 1) the smartphone associated to the AP, obtained its IP address via DHCP, and connected to the remote server. 2) the smartphone obtained its IP address via DHCP and connected to the local server. 3) the smartphone used static IP address configuration and connected to the local server. We repeated the experiments 10 times for each setting.

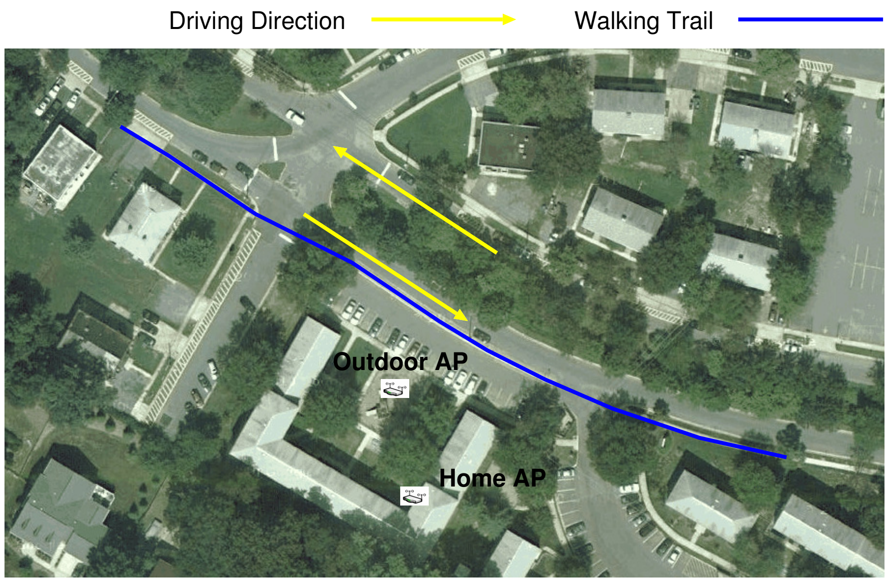


Figure 4.2: Map for east-west walking and driving experiments [16].

We plot the mean value of TCP downlink throughput and standard deviation in Figure 4.3. As we can see in the figure, we are able to improve TCP throughput by at least 200% when connecting to the local server instead of the remote server. The poor performance for the remote server case is caused mainly by the longer RTT and potentially the additional congestion-related losses on the Internet that impair TCP loss recovery notably. The impact of RTT and loss rate on TCP performance has been studied in [217, 218] which provide mathematical analysis to explain how losses and RTT affect TCP throughput. Although similar observations have been made by the studies of vehicular Internet access [85], another important benefit of separating the wireless links from wired ones is that *we can reduce the data transmission time, and thus save energy consumption on smartphones.*

The differences in throughput when walking in different directions may be caused by the location of home AP and the antenna direction of the smartphone. Thus, the duration of WiFi connectivity is ∼75 seconds when walking towards the east and ∼25 seconds longer (100 seconds) to the west. We note that as we conducted all the experiments in the wild, there are several other co-channel APs in the experimental area. Other users may use those APs for extensive data transfer during the experiments, which may interfere with the tests.

Regarding the impact of using DHCP, it is known that avoiding DHCP

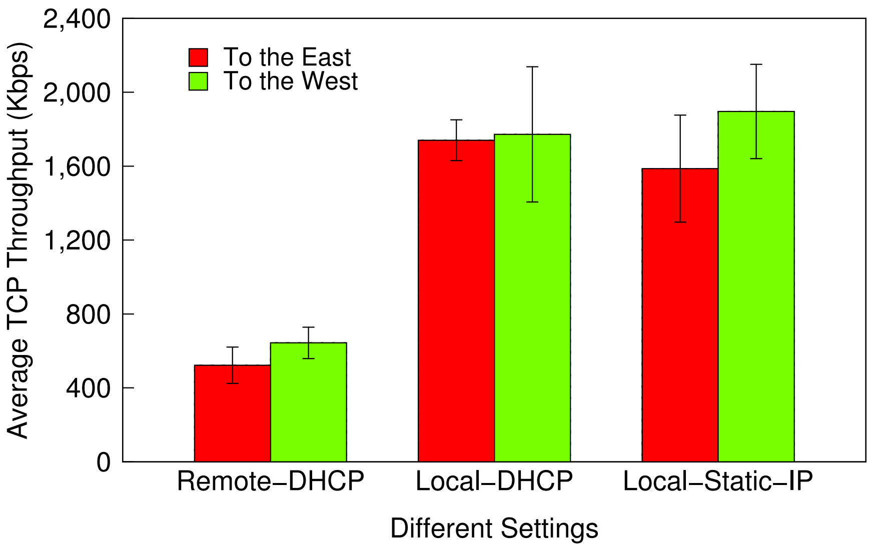


Figure 4.3: Downlink TCP throughput at walking speed [16].

can improve the performance of WiFi-based Internet access for mobile devices [257]. In Figure 4.3, we can see that the improvement of using static IP configuration (instead of DHCP) is not significant for the walking scenario. The main reason is that the saved setup time for IP address configuration is only around several seconds (less than 3 in our case), which is short compared to the duration of TCP connections (∼75 seconds when walking towards the east and 100 seconds for the west).

To further illustrate the difference between smartphones and notebooks, we measured the performance of smartphones and netbooks at driving speed. We did not use netbooks for the walking experiments because we do not consider it to be a typical usage scenario of netbooks.

The experimental setup and map are shown in Figure 4.1 and Figure 4.2, respectively, with the setting of local server and static IP configuration. We plot the sorted measurement results of the amount of transferred data and duration of TCP connections for 10 runs in Figure 4.4 and 4.5, respectively. These results are for the experiments of driving towards the west. When driving towards the east, although the smartphone was closer to the 'Home AP' in terms of physical distance, the device was placed near the driver (left-hand side of the car) and hence away from the right-hand car windows facing the AP. As the car itself may reduce the received signal strength, during our experiments, the smartphone could only associate with the AP, but failed to set up TCP connections.

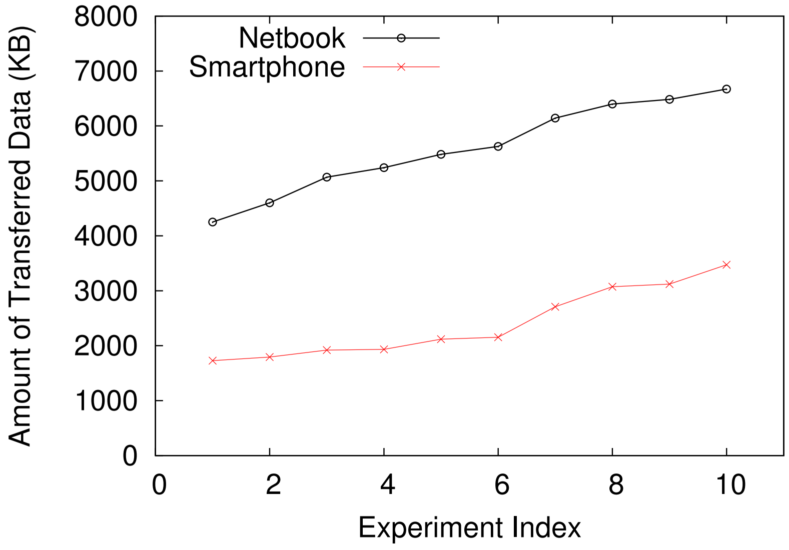


Figure 4.4: The amount of transferred data at driving speed [16].

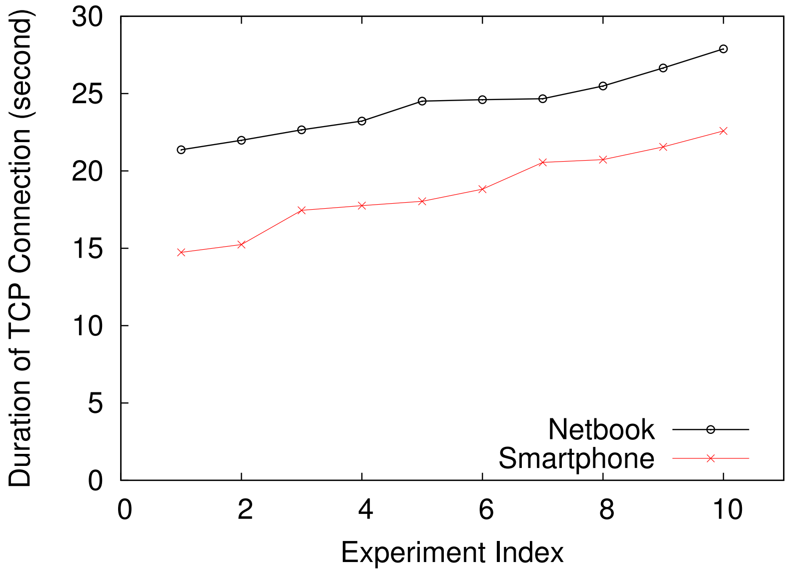


Figure 4.5: The duration of TCP connections at driving speed [16].

Table 4.4: Machines and antennas used in previous studies.

| Citation | Machine | Antenna |
|---|---|---|
| [5] | Hacom OpenBrick computer | 3 dBi gain omni-directional |
| [101] | CarTel embedded computer | 5.5 dBi gain omni-directional |
| [102] | Dell Latitude laptop | 12 dBi gain omni-directional |
| [257] | Dell D600 laptop | 7 dBi gain MA24-7N |
| [258] | Soekris net4511 computer | 15 dBi gain directional |

Although moving speeds may have less impact on 3G throughput, they do affect the performance of WiFi access, especially for smartphones. This is shown in Figure 4.4 and 4.5, as netbooks performed much better than smartphones. As vehicular WiFi access has been widely investigated, we check the previous work [5, 101, 102, 257, 258] to understand the performance gap between smartphones and netbooks. We summarize the machines and antennas used in their experiments in Table 4.4. With no exception, external antennas and PC-alike devices were used for their experiments, as shown in Table 4.4. It is well-known that the antenna plays an important role in WiFi-based Internet access. For example, Deshpande et al. [102] reported that a 12 dBi antenna provides better connectivity than 5 and 7 dBi antennas. Eriksson et al. [103] also found that mounting an external antenna on the roof of a car can significantly increase the signal strength of received WiFi frames. However, due to the size limitation, smartphones may not be able to use external antennas.

This finding matches our driving tests for smartphone and netbooks at driving speed. We note that the results in Figure 4.4 and 4.5 are for the local server connected to 'Home AP' over Ethernet. When connecting to the remote server (in Berlin), the smartphone can receive only several hundreds KB of data. Thus, mobile traffic offloading for smartphones is more challenging than that for PC-like netbooks.

### *Energy Cost*

To investigate energy cost in WiFi offloading, we measured the energy consumption of data transmission and offloading related operations. The experiments were conducted in Finland using Monsoon Power Monitor [3] with 5 KHz sampling rate. Figure 4.6 shows the equipment we used for energy measurement.

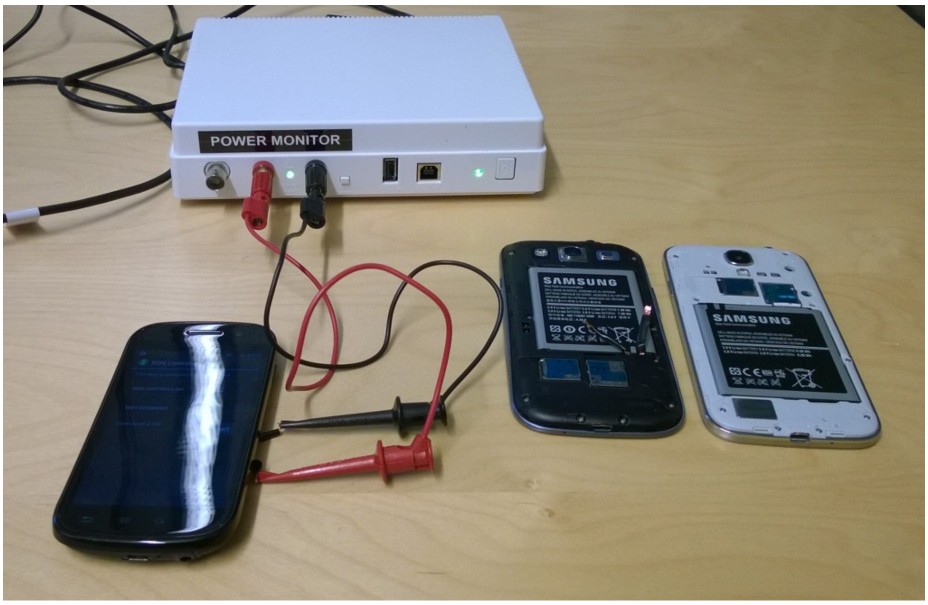


Figure 4.6: Equipment for energy measurement.

Regarding data transmission, we measured the energy consumption for transferring ∼20 MB data from a server to Nokia N900 and Samsung Nexus S smartphones through 3G and WiFi networks, using iperf TCP. We intentionally increased the distance between the smartphones and WiFi AP to measure the impact of WiFi access with poor signal quality. As we can see from the results in Table 4.5, when WiFi is of poor signal quality, TCP throughput decreases dramatically, resulting from the increase of wireless loss that undermines the TCP throughput [217, 218]. Since the throughput goes down, the system needs to work for much longer to transfer the same amount of data and it also needs to do extra work (e.g., transmitting more over WiFi) in order to recover from the losses. This inherently consumes more energy. In this case, data transfer through poor quality WiFi can consume more energy compared with the consumption under 3G [4]. *These results suggest that the solutions that utilize every offloading opportunity without considering the energy consumption may reduce the battery life of*

[3] `http://www.msoon.com/LabEquipment/PowerMonitor/`

[4] We provide additional measurements that demonstrate the effect of signal quality in Section 5.2

Table 4.5: Measured energy consumption of 20 MB data transfer.

| | N900 | | Nexus S | |
|---|---|---|---|---|
| | Energy Joule | Throughput Mbps | Energy Joule | Throughput Mbps |
| 3G | 109.4 | 1.89 | 65.40 | 1.99 |
| WiFi | 21.27 | 7.84 | 5.879 | 27.7 |
| WiFi (poor sig. quality) | **116.0** | **0.422** | **191.7** | **0.302** |

Table 4.6: Energy consumption (Joule) related to WiFi offloading.

| Device | $E_{W\_on}$ | $E_{W\_off}$ | $E_{asso}$ | $E_{scan}$ | $E_{GPS}$ |
|---|---|---|---|---|---|
| N900 | 0.18±0.025 | 0.13±0.021 | 0.28±0.13 | 0.53±0.077 | 4.0±1.3 |
| Nexus S | 0.27±0.019 | 0.29±0.016 | 0.25±0.049 | 0.27±0.017 | 10±1.3 |

*smartphones.*

To understand the operation overhead in WiFi-based traffic offloading, we summarize the energy consumption on N900 and Nexus S in Table 4.6. We repeated each measurement 10 times and reported the mean values and standard deviations. $E_{W\_on}$ and $E_{W\_off}$ show the energy consumption for turning on and off the WiFi interface. $E_{asso}$ shows the energy consumption for associating with a WiFi AP. $E_{scan}$ shows the consumption of a WiFi scanning in the laboratory environment. $E_{GPS}$ shows the energy consumption of AGNSS (Assisted Global Navigation Satellite System) location method with assistance data from external location server. One interesting observation from our WiFi scanning tests is that constant scanning may cause overheating on the smartphones, which makes the system unstable (i.e., application failure and system reboot).

### 4.2.2 Measurement insight

We investigated the following aspects through the measurement study: 1) Is it feasible to offload mobile data traffic to WiFi networks, given the current deployment and availability of WiFi APs in metropolitan areas? 2) Is the performance of WiFi-based Internet access for moving smartphones good enough to offload mobile traffic? 3) What is the impact of traffic offloading for smartphones in terms of energy consumption?

Our measurement study provides the following insight:

- For WiFi-based mobile data offloading, the number of open-accessible APs is very low, verifying the trend that less and less WiFi APs are open. Therefore, we propose *collaboration among cellular operators, WiFi service providers and end users to increase the offloading opportunities.*

- The performance of WiFi-based Internet access for moving smartphones is worse when connecting to remote servers than connecting to a local server, and is also worse than that of PCs (e.g., netbooks or laptops).
- The energy cost in WiFi-based offloading is visible. Using WiFi networks with poor signal quality for traffic offloading leads to high energy consumption on smartphones, thereby resulting in poor user experience. Meanwhile, we should avoid redundant WiFi scanning due to high energy cost.

To address these challenges, we need a *collaborative design*, which can bring together cellular operators, WiFi service providers and end users to increase the opportunities to offload mobile traffic to WiFi. To increase the amount of offloaded mobile traffic and save energy consumption on smartphones, we need optimization schemes such as to *prefetch predictable data traffic at WiFi APs.* In order to enable smartphones to select the most energy efficient WiFi APs to offload mobile traffic, we need a deployable and effective energy-aware offloading algorithm to assist the offloading decision.

We note that there are similar studies in this area using PC-like devices [5, 85, 102]. Our measurement study and answers to these questions lay the foundation for the proposed collaborative MADNet energy-aware WiFi offloading dedicated for ***smartphones***.

## 4.3 System Design

The proliferation of smartphones and data-intensive mobile applications has created pressure on the limited capacity of cellular networks. We advocate that mobile traffic offloading is a 'win-win-win' strategy for cellular operators, WiFi service providers, and end users, and that they should cooperate to make it feasible and effective. With offloading solutions, cellular operators can meet the rising traffic demand without significantly increasing their CAPEX (capital expenditure) and OPEX (operating expenses). Moreover, they can provide traffic offloading as a value-added service, and thus expand their user base. The collaborative offloading schemes may also bring third-party WiFi service providers more customers without service contracts, and thus extra revenue. Finally, end users can benefit from

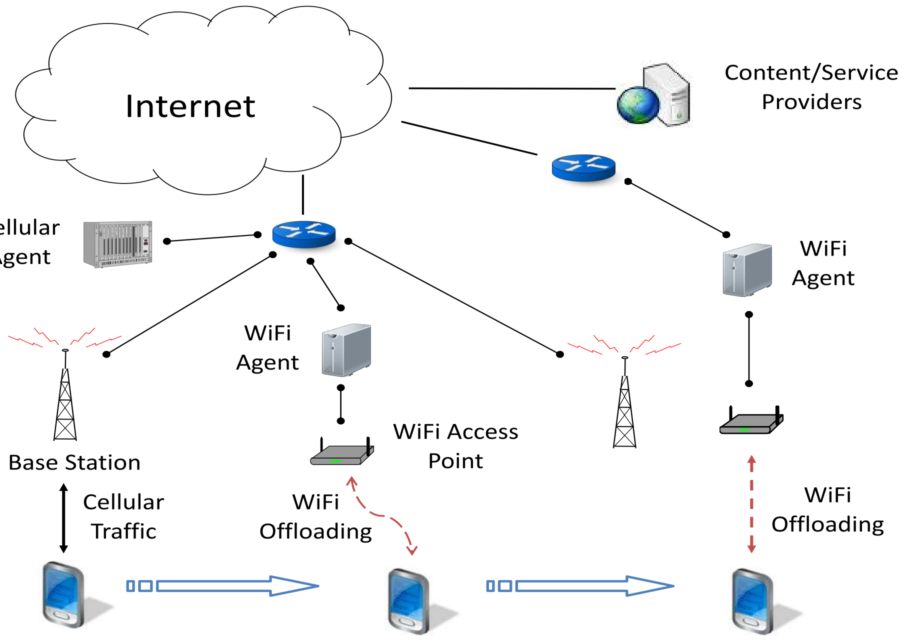


Figure 4.7: Overview of MADNet energy-aware offloading.

their participation through higher data rate and longer battery life for their smartphones.

Figure 4.7 presents an overview of our proposal, in which smartphones are guided by MADNet to offload cellular traffic to collaborative WiFi networks. We aim to design an energy-aware approach to offload streaming traffic for smartphones that are ***in motion***. When people walk, although they usually do not read news or watch video, they often listen to music. In addition, more and more people watch video using smartphones when riding on public transport services such as metro and buses. According to Cisco Visual Network Index [1], nearly 3/4 of the global mobile data traffic will be video streaming by 2019. Between 2014 and 2019, there will be a 13-fold increase for mobile video traffic, which accounts for 72 percent of total mobile data traffic by the end of the forecast period. Although real-time streaming traffic can not tolerate much delay, it is feasible to predict the delivery of *non-live streaming* traffic, such as Spotify [5] and YouTube [6]. When mobile users listen to music online, they usually make a playlist to include all the songs they like. With the playlist, we can know the expected streaming traffic for the next several minutes. The same principle is true

[5] https://www.spotify.com/
[6] https://www.youtube.com/

for video streaming. Once a mobile user starts to watch video online, the future streaming content becomes predictable. Therefore in our system design, smartphones may start data streaming immediately over cellular networks and meanwhile try to find potential WiFi APs that can deliver streaming data with lower energy cost.

### 4.3.1 Collaborative mobile traffic offloading

Generally, there are three types of WiFi APs: commercial hotspots, open-access hotspots, and home APs deployed for private use. Nowadays, there are a large number of WiFi hotspots deployed by cellular network operators and third-party WiFi service providers. Besides their own deployed hotspots, cellular network operators can sign service contracts with other WiFi service providers (e.g., Boingo [7] to serve their customers. End users themselves can also participate in mobile data offloading by sharing their home APs with others [259]. Moreover, there are companies like FON [8] that enable the sharing of WiFi Internet connections by creating both private and public networks on a wireless router. For many home APs, the Internet service providers are also the cellular network operators.

Given that the open-accessible APs for WiFi offloading is very low (Section 4.2.1) and there are potentially a large number of deployed WiFi APs, it is important to enhance collaboration among cellular network operators, WiFi service providers and end users to create more opportunities to use WiFi for offloading. We hence propose a collaborative offloading design that intelligently aggregates the collaborative power of cellular operators, WiFi service providers and end users, and integrate energy awareness at the architecture level. It has the following three key design goals:

- Offloading the mobile data traffic for smartphones, which is enabled by the *collaborative* architecture.
- Making the offloading process *transparent* to end users.
- Reducing the *energy consumption* of smartphones, which is realized by the energy-aware offloading decisions.

Figure 4.8 illustrates the major components of our proposal, referred to as MADNet: a client module on smartphones (`Smart-Client`), an agent on WiFi APs (`WiFi-Agent`), and another agent in the cellular access networks

[7] `http://www.boingo.com`
[8] `https://corp.fon.com/en`

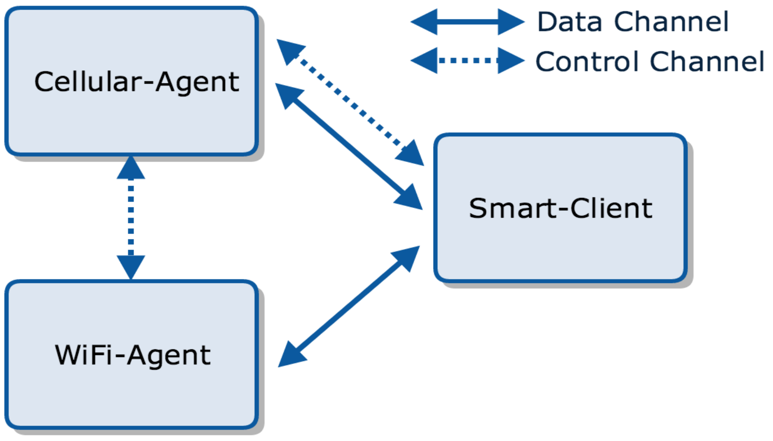


Figure 4.8: MADNet components.

(`Cellular-Agent`). To coordinate the split of data traffic between these two networks, we set up two control channels, as shown in Figure 4.8.

MADNet makes offloading decisions based on the contextual information of smartphones and knowledge-base of operators. First, `Cellular-Agent` knows the locations of the neighboring WiFi APs for a given smartphone according to its current position, because these locations are fixed (and known through services such as `wigle.net`) and some WiFi APs are deployed by cellular operators. Second, as people are creatures of habit and usually take similar paths every day, it is feasible to predict their mobility patterns using history information [7, 86, 216].

When a mobile user requests content from the Internet, `Smart-Client` will send the geographical location, moving speed, direction, content list to `Cellular-Agent`. `Smart-Client` can retrieve this information from the wireless interface and various sensors (e.g., WiFi, GPS, and accelerator). By following the common practice of “Wake on Wireless” [179] and thus reduce energy consumption of smartphones, `Smart-Client` turns on the WiFi interface and the sensors only when users issue content requests.

`Cellular-Agent` utilizes the proposed offloading decision algorithm (Section 4.3.3) to determine whether offloading cellular traffic to a given WiFi AP can potentially save energy on smartphones. It notifies `WiFi-Agent` to prefetch the content if a candidate AP is found. With the context information from `Smart-Client` as input, the heavy computational tasks, including positioning and mobility prediction are hence offloaded to `Cellular-Agent`.

After receiving information from `Cellular-Agent` about the WiFi AP (e.g., MAC address, ESSID and time to associate) for offloading, `Smart-Client` can initialize the WiFi association as instructed by `Cellular-Agent`, and download prefetched data via the target MADNet-enabled AP.

### 4.3.2 Optimization for smartphones

Given the data transfer performance observed in our war-driving and war-walking experiments (Section 4.2.1), it is challenging for smartphones to use WiFi networks to offload data traffic even at walking speed (i.e., for pedestrians). MADNet adopts several mechanisms dedicated for smartphones to improve the performance in WiFi offloading.

#### *Content Prefetching*

We use content prefetching to improve the wireless access performance. The effect of content prefetching has been studied and considered suitable for mobile devices [189]. The prefetching approach is important especially for traffic offloading on smartphones, given the limited size of smartphones where internal antennas can restrict the wireless access performance.

Since our system focuses on mobile streaming, we leverage the predictable feature of streaming content to achieve effective prefetching. In MADNet, the `Smart-Client` manages the content play lists for the mobile user. Once the location context is ready and there is a demand for streaming services, `Smart-Client` will convey this information to `Cellular-Agent`. By selecting a suitable MADNet-enabled AP for offloading, `Cellular-Agent` will push the play list to the target AP on which `WiFi-Agent` is installed. `WiFi-Agent` will prefetch the pending content in the list from the Internet and then serves the prefetched content once the smartphone successfully connects to this AP. We note that the overhead of prefetching can increase if the prediction is incorrect [189], making the content fetched by a WiFi AP redundant. To minimize the negative impact of inaccurate prediction, `WiFi-Agent` will remove the prefetched content in its local storage according to a pre-defined time threshold, and thereby saves storage space.

#### *Positioning*

To identify the target AP for traffic offloading, MADNet utilizes the user location as a key context to make offloading decisions. To be energy efficient, MADNet leverages different technologies to provide location information under different environment contexts. For example, GPS is regarded as power-hungry and can drain the battery of mobile phones in a few hours [260]. Therefore, MADNet utilizes a WiFi positioning scheme to save energy if such context is available (i.e., in commercial areas with WiFi coverage). To make our system versatile and capable for different environments, `Smart-Client` will use GPS if the WiFi positioning scheme can not locate the end user. Regarding GPS positioning, it takes 14 seconds on the average for a cold-start GPS to get the first accurate fix using AGNSS according to our measurement on N900. For the location method using only internal GPS (i.e., GNSS), the cold-start duration on N900 is around 20 seconds with 6.30 Joules of energy consumption. We hence prefer AGNSS for positioning in MADNet.

The WiFi-based positioning scheme in MADNet is based on RADAR [77], which uses WiFi fingerprints with reference radio-map. `Smart-Client` conducts WiFi scanning three times at the beginning to collect three groups of samples. It then filters the scanning results by voting, so that only those APs that appear at least twice in the results are kept. `Smart-Client` then sends the aggregated samples to `Cellular-Agent` via cellular connection. `Cellular-Agent` compares the sampled values to its WiFi fingerprint database and selects a location with the highest similarity as the estimated user location.

We note that a single WiFi scanning sometimes exhibits arbitrary results regarding signal strength and number of APs nearby. This affects the accuracy of location estimation. MADNet hence demands `Smart-Client` to conduct three WiFi scannings to assist the positioning process. The overhead of WiFi scanning for three times is obviously higher than scanning once, but compared with the impact of incorrect positioning, which may result in missing the opportunity to use a WiFi AP for offloading, our method aims to strike a balance between the positioning accuracy and scanning overhead.

#### *Mobility and Target AP Prediction*

Selecting a suitable WiFi AP for offloading requires the MADNet system to predict the mobility of end users. Inspired by the K Nearest Trajectories (KNT) algorithm that uses history trace to predict vehicle mobility [7], we design a prediction algorithm for MADNet, Average Nearest Trajectories (ANT), to help select a suitable WiFi AP for offloading.

Given that the basic KNT algorithm only utilizes historic traces (i.e., position of a user in the previous 20 seconds or 1 minute) to predict future location, it does not take into account the regularity of human mobility

[216]. Our ANT algorithm is dedicated for smartphones in the traffic offloading scenario. The major steps of ANT are illustrated as follows:

- Determine the average nearest trajectories: ANT utilizes an established database for location coordinates that reflect the mobility history of end users (e.g., collected by cellular operators from collaborative users with energy-efficient technique [211]). Each entry inside the database contains a pair of values ($T$, $C$), where $T$ is the recorded time in a day formatted as "Hour:Minute:Second", and $C$ is the coordinates of the recorded spot. A trajectory in ANT is a number of consecutive entries in the database following the time-line sequence, such as [($T_1$, $C_1$), ($T_2$, $C_2$) .. ($T_n$, $C_n$)].

  MADNet utilizes WiFi/GPS positioning to infer the current location, which serves as an input for the ANT algorithm. By calculating the distance of the estimated current location against a set of coordinates in historic trajectories ($HT$), ANT seeks the average nearest trajectories in the database, which are essentially a set of trajectories that have the shortest average distance from the estimated current location to all the entries on each trajectory. We then select those average nearest trajectories (i.e., nearest neighbors) for mobility and target AP prediction.

- Predict the future mobility and vote for a candidate AP: Based on the average nearest trajectories determined in the previous step, ANT conducts a linear search in each trajectory to find an entry ($T$, $C$) with the closest $T$ matching the current time of the day. After finding such an entry in a trajectory, ANT infers the future mobility based on the trajectory where this entry is located. By using a WiFi location database containing the coordinates of MADNet-enabled APs, ANT uses the selected entry as a starting point to check the next entries on the trajectory that are within a time interval (e.g., by considering user's moving speed). ANT then infers a group of APs visited by the end user along the trajectory within that interval. A WiFi AP is determined to be a visited one when its distance to a location on the trajectory is within a pre-defined range. ANT adopts a voting process on the selected APs to check whether they have been visited by at least half of the average nearest trajectories. We then report all the APs that fulfill such requirements as the candidate APs.

This algorithm differentiates from KNT in a couple of respects. First, we use the current location only, while KNT assumes the device constantly records its previous GPS location, which is inefficient in terms of energy consumption. Second, we use more efficient linear search rather than linear interpolation because once we have determined the average nearest trajectories, the future mobility trajectories can be deduced from the database by considering the regularity of human mobility [216]. Finally, the goal of ANT is to find the suitable WiFi APs for offloading. The candidate APs will serve as input for the energy-aware offloading algorithm (Section 4.3.3) to select a final WiFi AP which is the most energy efficient one to offload mobile traffic.

The main reason we design ANT for MADNet is for the fact that smartphones can not afford continuous WiFi scanning due to the high energy cost. Enabled by the collaborative design (e.g., operator's knowledge on AP location, mobility history of end users), MADNet only requires a few scanning samples for positioning. Based on such location input, ANT suggests a group of candidate APs that will be potentially visited by the end user. We note that with the limited number of location input, MADNet may suffer from prediction inaccuracy especially when an end user is at a crossroad. One approach is to use mobile sensors to obtain orientation context to compensate the limited number of positioning samples, which can be integrated on the `Smart-Client`.

Since mobility prediction is a challenging task [261], we aim to minimize the negative impact when a prediction is incorrect. In MADNet, even if the prediction is inaccurate, `Smart-Client` can detect that there is no suitable AP to associate with (e.g., WiFi association failed). In such a case, `Smart-Client` continues to use cellular connection to fetch streaming content, without interrupting the ongoing data flows.

### 4.3.3 Energy-aware offloading design

We propose an energy-aware offloading algorithm to assist MADNet in making offloading decision. As illustrated in Algorithm 1, MADNet utilizes candidate APs $AP_{set}$ suggested by ANT algorithm as the input for the energy-aware offloading algorithm to select the most energy efficient WiFi AP $AP_{target}$ for traffic offloading. For each execution of the algorithm, the maximum saving $Save_{max}$ is initialized to 0 and $AP_{target}$ is set to 'NULL'.

Regarding data transmission, WiFi is generally more energy efficient than 3G [130], but offloading mobile traffic to WiFi introduces extra energy consumption, such as to obtain location information and to associate with the WiFi APs that are predicted to be available. MADNet performs traffic offloading only when using WiFi for transferring data (instead of through 3G networks) saves more energy than the extra energy consump-

**Algorithm 1** Energy-Aware Offloading Algorithm.

```
Require: The candidate APs $AP_{set}$ predicted by ANT algorithm.
Require: The power of data transfer $P_{3G}$ for 3G and $P_W$ for WiFi.
Require: The head and tail energy $E_T$ of 3G and $E_{oo}$ for the offloading
    related overhead.
 1: for each $AP \in AP_{set}$ do
 2:   Predict the throughput $B_{3G}$ for 3G network and estimate the off-
      loading capacity $C_W$ and throughput $B_W$ of this $AP$.
 3:   Predict the prefetching capability $F$ of this $AP$.
 4:   Calculate the WiFi offloading duration $C_W/B_W$ and the time to re-
      ceive the same amount of data through 3G network $C_W/B_{3G}$.
 5:   if $F \geq C_W$ and the following inequality holds

          $E_T + P_{3G} \cdot C_W/B_{3G} > k \cdot E_{oo} + P_W \cdot C_W/B_W$      (4.1)

      then
 6:     if the saving gap $(E_T + P_{3G} \cdot C_W/B_{3G} - k \cdot E_{oo} - P_W \cdot C_W/B_W)$
        is greater than the maximum saving $Save_{max}$
        then
 7:       Update $Save_{max}$ and $AP_{target}$ for this $AP$
 8:     end if
 9:   end if
10: end for
11: Offload mobile data traffic to the selected $AP_{target}$.
```

tion overhead. For energy saving, `Smart-Client` only scans WiFi when it is necessary (i.e., guided by the `Cellular-Agent`).

We describe this requirement rigorously in the inequality (4.1) of Algorithm 1, where $k$ is a parameter to accommodate measurement errors. For small values of $k$, the estimation errors may cause more energy consumption on smartphones due to offloading. On the other hand, we may lose some offloading opportunities if $k$ is too large. For experiments, we set $k$ to be 1.1 tentatively in order to investigate the gains of WiFi offloading.

The energy-aware offloading decision is affected by the throughput of 3G and WiFi networks. For the measurement study of Wiffler [5] in Amherst, the downlink median TCP throughput is 600 Kbps for 3G and 280 Kbps for WiFi. In this case, although offloading 3G traffic to WiFi networks can reduce 3G usage, it may cause more energy consumption on smartphones, as indicated in our measurement study (Section 4.2.1). In another measurement study by Deshpande et al. [102], WiFi offers substantially higher median throughput than 3G, ~2000 Kbps vs. ~500 Kbps, respectively [9]. For this scenario, WiFi-based offloading may potentially reduce the energy consumption of smartphones.

For calculating the energy saving, we need to know the predicted throughput of the 3G network $B_{3G}$ and the WiFi offloading capacity $C_W$ (i.e., the number of bits we can offload to WiFi). Through an eight-month measurement study, Yao et al. [124] show strong correlation between cellular throughput and location for 3G HSDPA networks, and thus it is feasible to estimate 3G throughput for a specific location (e.g., WiFi offloading area) by using the history record of 3G throughput of the location. As pointed out in Wiffler [5], we can also estimate the offloading capacity and throughput of WiFi networks using existing work like BreadCrumbs [86].

Because MADNet utilizes prefetching to assist WiFi offloading, the maximum prefetching capability $F$ for a WiFi AP is defined as the product of the prefetching duration and the available backhaul throughput of this WiFi AP (i.e., from server over Internet to AP). The prefetching duration is the period from the moment `WiFi-Agent` starts to prefetch content according to the notification from `Cellular-Agent` till the expected time `Smart-Client` dissociates from the WiFi AP (i.e., moving away from the WiFi coverage). The value of $F$ is maintained by `Cellular-Agent` for each MADNet-enabled WiFi AP. `Cellular-Agent` pulls such information from `WiFi-Agent` after every offloading session and records it for each specific `Smart-Client`. For bootstrapping, MADNet sets $F$ to a pre-defined maximum value. To better utilize the capacity of a WiFi link, the result of the offloading decision will be negative if this prefetching capability $F$ is smaller than the estimated offloading capacity $C_W$. In this case, MADNet will still prefer using cellular connection.

To calculate the saved energy, we estimate the $C_W/B_{3G}$ and $C_W/B_W$. For bootstrapping, we use pre-defined historic values for $C_W$, $B_{3G}$, and $B_W$. MADNet updates those values by collecting them regularly from `WiFi-Agent` and `Smart-Client` after every offloading session. Due to the radio resource control of cellular networks, after transmitting or receiving a packet the 3G radio stays at high power state and drops to low power only when the interface has been inactive for several seconds [130]. This state transition also introduces significant head (from low power to high at the beginning) and tail (from high power to low at the end) energy, which is considered as $E_T$ in (4.1) of Algorithm 1.

[9]The contradictory results reported in the above studies may be caused by the fact that Wiffler [5] used an 802.11b radio for all the experiments, which limits the maximum PHY bitrate to be 11 Mbps.

Since it is important to keep the existing data flows uninterrupted to guarantee consistent service experience in traffic offloading, MADNet checks the workload status on `Smart-Client` for both offloading initiation and termination. For data flows sensitive to connectivity interruption, MADNet follows the principle in the initiation phase to keep using the existing channel (e.g., 3G) until the ongoing flows complete and then offload new flows to the new channel. For flows that can tolerate connectivity interruption, MADNet postpones the data transmission for a pre-defined delay tolerance threshold and then offloads the ongoing flows to the new channel once the connectivity is established. To terminate the offloading connection and switch back to 3G, MADNet adopts the 'fast switching' technique [5], which enables efficient connectivity switch to 3G when facing poor WiFi condition. In this regard, the prior research on handover mechanisms have provided guidance for seamless migration from one access network to another [55], [63]–[68], [81]–[84], which are valuable supplements to MADNet.

We note that our approach assumes that 3G and WiFi interfaces keep at the same data reception power level during the data transfer. This is reasonable for 3G interfaces but conservative toward WiFi as the moving speed of an end user affects the power level of WiFi interfaces [19]. Our approach relies on the collected average values of $C_W$ and $B_W$, and mainly provides an estimation for the actual energy saving.

In a nutshell, MADNet avoids the offloading of mobile data traffic to WiFi networks in poor condition and enables smartphones to select the most energy efficient WiFi AP if possible. It adopts data prefetching at WiFi APs, which can further improve the utilization of WiFi access, thereby reducing energy consumption on smartphones. We also note that the performance of streaming applications is not affected by these offloading decisions, because when offloading is not possible or feasible (e.g., caused by incorrect mobility prediction) MADNet relies completely on cellular connection for data transfer.

## 4.4 Implementation and Experiments

We develop a prototype for our MADNet energy-aware offloading to explore the benefits of WiFi-based traffic offloading for smartphones, and evaluate its performance with smartphones *in motion*.

### 4.4.1 System implementation

Our system consists of three core entities: `Cellular-Agent`, `WiFi-Agent`, and `Smart-Client`. The communication and schematics of the imple-

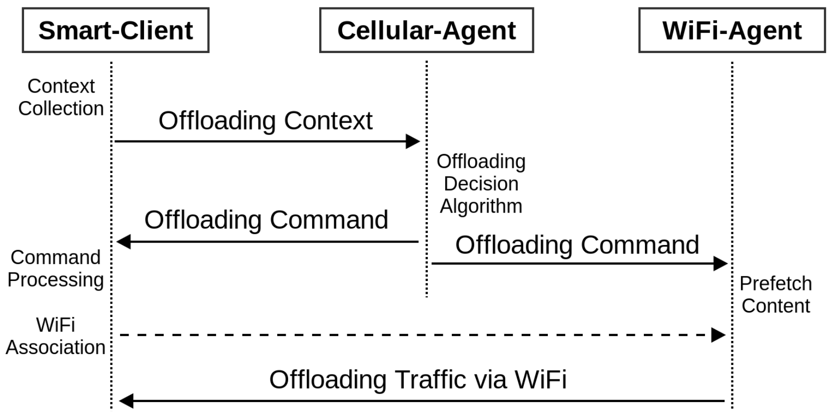


Figure 4.9: Schematics of MADNet system.

mented system is shown in Figure 4.9

### *Cellular-Agent*

The major duties of `Cellular-Agent` include to estimate the location and predict the mobility of end users, to make offloading decisions, and to forward content requests to `WiFi-Agent` and coordinate the authentication between smartphones and APs if necessary. We implement a WiFi-based positioning scheme similar to the standard RADAR approach [77] as described in Section 4.3.2. For mobility and AP prediction, we implement the proposed ANT algorithm by considering the regularity of human mobility. The output of the ANT algorithm is a group of WiFi APs that are predicted to be visited by `Smart-Client`. `Cellular-Agent` runs the energy-aware offloading algorithm described in Algorithm 1 to select a WiFi AP that can achieve the maximum energy saving. `Cellular-Agent` then sends the MAC address and ESSID of the candidate AP to `Smart-Client`. If the predicted WiFi-AP set is empty or none of the WiFi APs can satisfy the requirements specified in Algorithm 1, `Cellular-Agent` will notify `Smart-Client` to keep using cellular access. The `Cellular-Agent` module is deployed on a server machine located in our research laboratory with a public IP address for Internet access.

### *WiFi-Agent*

We implement `WiFi-Agent` as a service daemon in C++ and deploy it on a Linux system equipped with WiFi interface, which serves as a MADNet-enabled AP in our experiments. Once `WiFi-Agent` starts, it connects to `Cellular-Agent` and follows the offloading commands to support the offloading process. `WiFi-Agent` will perform data prefetching from the suggested Internet service and upload the information regarding each offloading session to `Cellular-Agent`. As instructed by `Cellular-Agent`, `WiFi-Agent` saves the prefetched content to its local memory and feeds them to `Smart-Client`. The amount of data to prefetch is determined by the estimated WiFi offloading capacity. As we focus on streaming applications, we implement the core functionality of `WiFi-Agent` optimized for downlink traffic in the current prototype.

#### *Smart-Client*

We implement a music streaming application with `Smart-Client`, called `MStreamer`. Once a playlist is given by a mobile user, `MStreamer` will start to stream the first music over a 3G network. At the same time, `Smart-Client` sends the context information to `Cellular-Agent` to make the offloading decision. By design, both WiFi and GPS can be used to obtain positioning context. For energy saving, `Smart-Client` prioritizes WiFi over GPS and will use GPS only if WiFi positioning fails. If `Cellular-Agent` chooses to offload traffic to WiFi networks, it will notify `Smart-Client` to submit the whole playlist and then forward the prefetching instructions to `WiFi-Agent` running on the WiFi AP that the smartphone will visit soon. Once the smartphone enters the coverage area of this WiFi AP (determined by a timer set by `Cellular-Agent` based on mobility prediction), it associates with the target AP and starts to download the prefetched songs into its local buffer. The `MStreamer` can then play the next music directly if it is in the buffer and avoid the streaming over 3G networks. We deploy `Smart-Client` on a Nokia N900 and use Jamendo [10] as the online music service for our experiments.

Although the cooperative architecture has several requirements on WiFi APs (e.g., storage space to prefetch data), we note that it is feasible on modern APs which are programmable and have USB ports to attach storage devices. For example, the NaDa system [262] has been proposed to leverage the computing and storage capabilities on ISP-controlled home gateways to reduce energy consumption of Internet-based video streaming services.

[10] `https://www.jamendo.com/`

### 4.4.2 Experimental evaluation

Our experimental evaluation consists of three parts: 1) We measure the streaming performance to show how much we can offload mobile traffic for smartphones at driving and walking speed. 2) We evaluate the impact of data prefetching and prediction schemes in MADNet. 3) We further estimate the energy savings achieved by our proposal for music streaming applications. We use the power measurement data obtained from the laboratory, since it is hard to measure directly the energy consumption for moving smartphones.

The driving and walking tests were carried out by Dr. Bo Han in College Park, Maryland, USA. To evaluate the full system, the author further evaluated the prototype implementation through field tests along the Limingantie street in Helsinki, Finland. In the evaluation, we focus on the case that each `WiFi-Agent` serves one `Smart-Client` at a time to investigate the effectiveness of our proposal.

#### ***Test Setup for Driving and Walking Experiments***

The first setup and test trail in College Park USA for the driving and walking experiments are depicted in Figure 4.2 (similar to the layout discussed in Section 4.2.1). For performance evaluation, we used another 'Outdoor AP' running on a netbook as a MADNet-enabled AP, which was placed ∼2 meters above the ground and ∼18 meters away from the road. We used this AP to evaluate the typical scenario of roadside APs, which are very close to the road. All the content has been pre-loaded to the AP so that `Smart-Client` can fetch directly from `WiFi-Agent` installed on the AP.

#### ***Test Setup in Helsinki***

Regarding the setup in Helsinki for evaluating the full system, Figure 4.10 shows the area map where we conducted outdoor experiments. We select a street, Limingantie, in a residential area near the campus. The conditions of this street including street length and WiFi coverage (148 APs detected in this area) make it suitable for evaluating the ANT and positioning schemes used in MADNet. We placed the MADNet enabled AP in Limingantie 42, which is about in the middle of the test trail. The AP is placed ∼1 meters above the ground and ∼5 meters away from the street.

Figure 4.11 shows the test equipment, including a Nokia N900 smartphone installed with `Smart-Client`. The smartphone connected via 3G

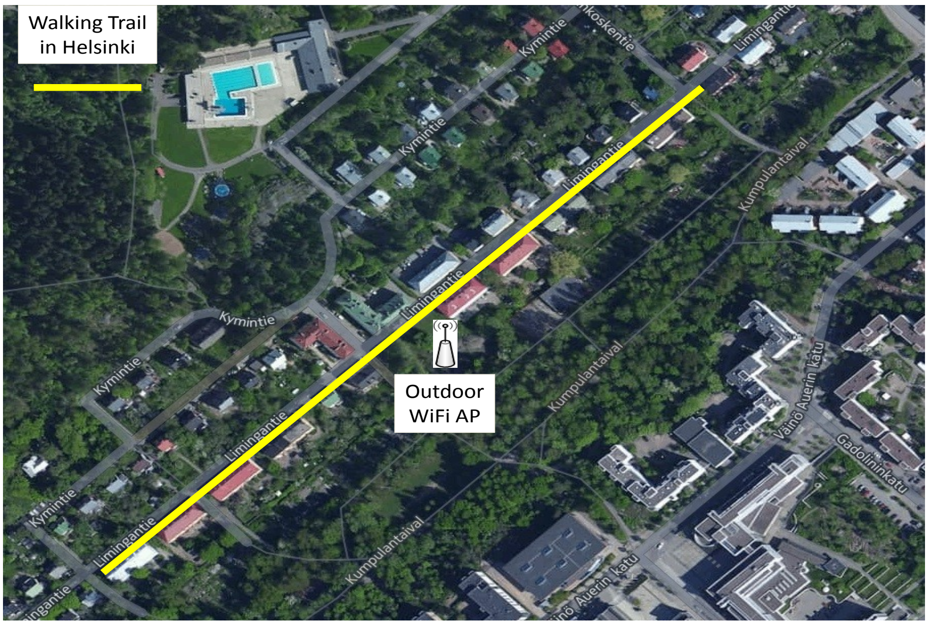


Figure 4.10: Map for outdoor experiments in Helsinki.

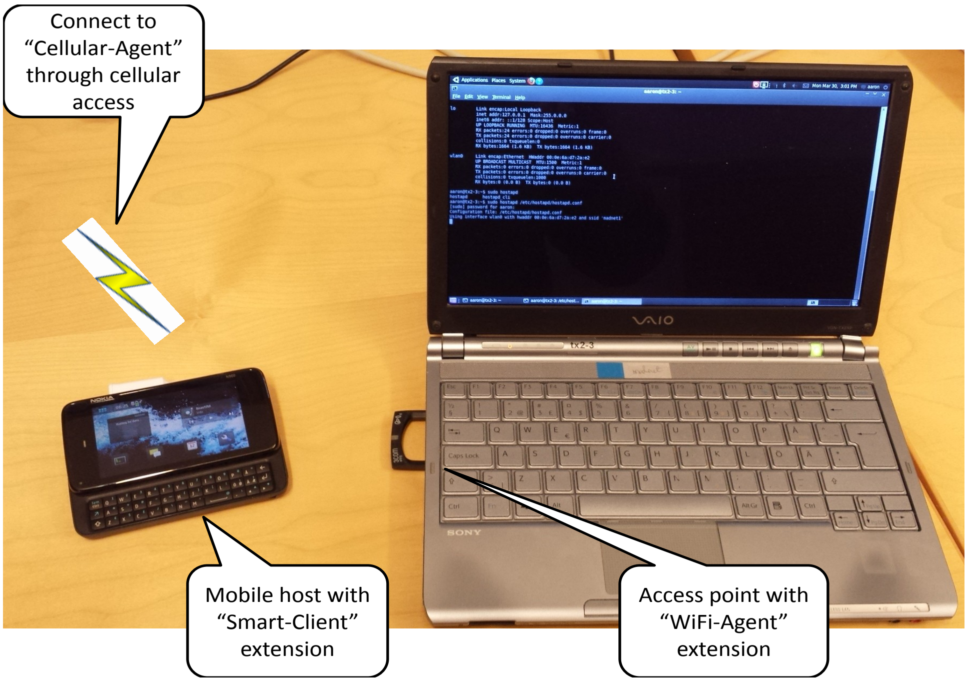

Connect to "Cellular-Agent" through cellular access
Mobile host with "Smart-Client" extension
Access point with "WiFi-Agent" extension

Figure 4.11: Test equipment for live network experiments.

to our server in the laboratory running `Cellular-Agent`. We installed `WiFi-Agent` on a laptop machine, equipped with an external PC card with a XJACK WiFi 802.11 a/b/g antenna, manufactured by 3Com (model 3CRPAG175 with chipset AR5212).

***Driving vs. Walking in WiFi Offloading***

The measurement results in Figure 4.3 (Section 4.2.1) already indicate that placing content at a local server can improve the downloading performance for smartphones at walking speed. We further compare the amount of streaming data fetched from an outdoor AP at both driving and walking speeds, considering both TCP and HTTP protocols. This experiment explores the case when MADNet-enabled AP has prefetched the content. The results show how much our system can offload using TCP and HTTP protocols.

Figure 4.12 and 4.13 show the amount of streamed data and the total playing time of these streamed songs, using the HTTP protocol. We conducted the experiments 10 times and report the mean values with standard deviations. As we can see in these figures, when driving towards the west, `Smart-Client` can receive on average ∼4.6 MB streaming data from the WiFi AP, with average playing time of ∼400 seconds. For the walking scenario, the offloading performance is much better than driving.

We note that when driving to the east, although the smartphone was closer to the 'Outdoor AP' in terms of physical distance, the device was placed near the driver (left-hand side of the car) and thus away from the right-hand windows of the car facing the AP. Since the car itself may reduce the received signal strength, the smartphone could only associate with the AP, but failed to fetch the content via HTTP in the experiments. Such an effect is also visible in the results of walking scenarios, as the performance of walking to the west is better than that of walking to the east. The results indicate that WiFi offloading is challenging for smartphones in a vehicle at driving speed, even when MADNet can prefetch data to the local server.

To investigate the protocol overhead of HTTP, we also plot in Figure 4.14 the amount of transferred streaming data through TCP and HTTP from the 'Outdoor AP', with the smartphone moving at driving speed. We repeated the experiments 10 times for each setup and plot the sorted results.

Due to the same reason stated above, `Smart-Client` was not able to fetch streaming content from `WiFi-Agent` using HTTP when we drove towards the east. Meanwhile, `Smart-Client` can receive data using TCP

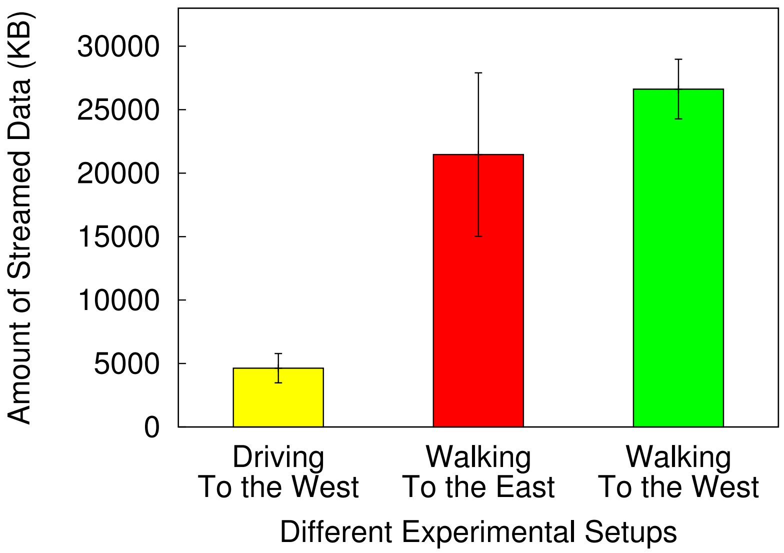


Figure 4.12: The amount of streamed data at driving and walking speeds [16].

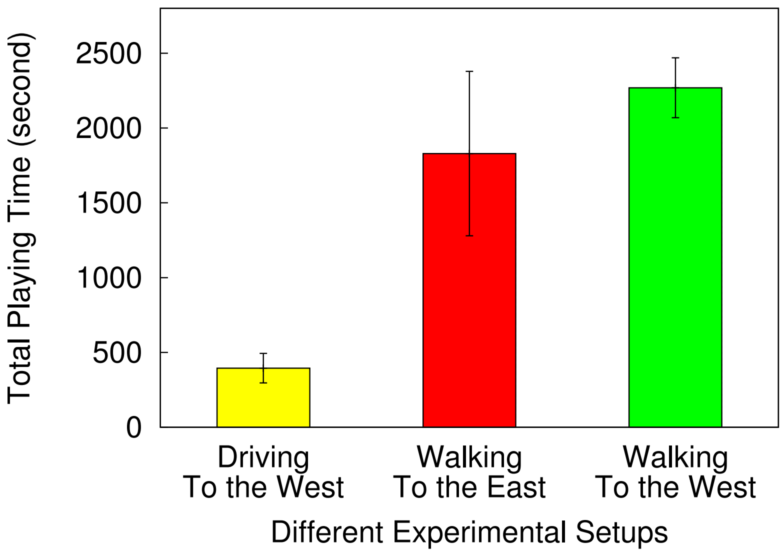


Figure 4.13: The total playing time of streamed songs at driving and walking speeds [16].

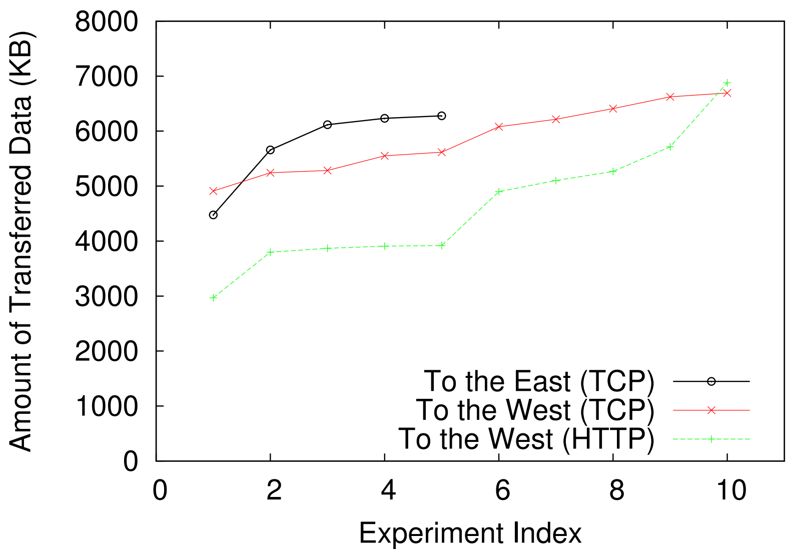


Figure 4.14: The amount of transferred data at driving speed [16].

from `WiFi-Agent` for 5 experiments when driving towards the east. This figure reveals the overhead of HTTP. In particular, when using the HTTP protocol the average amount of streamed data is about 80% (4,633 vs. 5,863 KB) of that downloaded through TCP, due to its protocol overhead.

***Content Prefetching & Predicted Locations for Association***

Given the low throughput of WiFi offloading at driving speed, we choose to evaluate our system prototype at walking speed in Helsinki. In order to enable positioning and prediction schemes, we built a WiFi fingerprint radio-map for the test trail as shown in Figure 4.10. All the reference locations were derived from real map to obtain accurate coordinates. We collected the WiFi scanning samples along the street and constructed a simple WiFi positioning database with 149 APs detected in this area (including our own MADNet-enabled AP). Each entry in the database contains the location of each reference point and a list of APs with their corresponding signal strength.

For the prediction schemes, we built a historic trajectory database using collected GPS and WiFi scanning traces in the past few days before the experiment along the street *Limingantie*. Each entry in the database contains the coordinates of location along the street, its recorded corresponding time, and a list of WiFi APs detected at this spot.

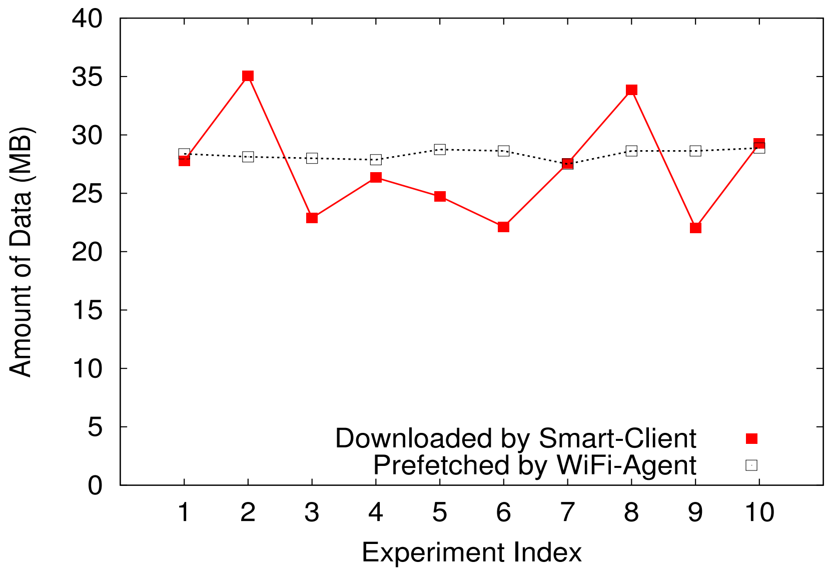


Figure 4.15: Offloaded and prefetched volume for walking towards east [16].

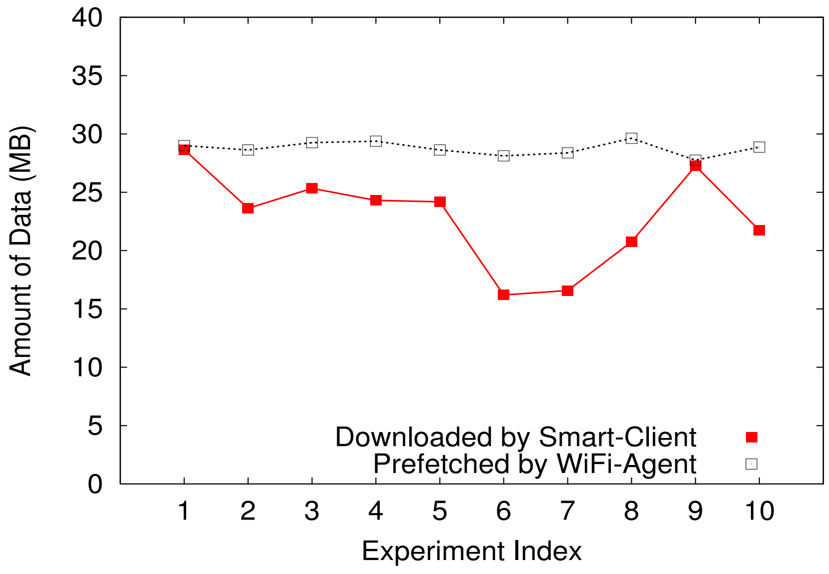


Figure 4.16: Offloaded and prefetched volume for walking towards west [16].

For bootstrapping the energy-aware offloading algorithm, we measured the 3G throughput in the test area and used the measured value 765.436 Kbps for $B_{3G}$. We also measured the offloading capacity for $C_W$ and WiFi throughput $B_W$ by downloading music data from the target WiFi AP located in Limingantie 42. The offloading capacity for walking towards east is 20.65 MB, and 18.5 MB for walking to the west. The throughput for walking towards east is 2.3 Mbps and 2.1 Mbps for walking to the west. The predicted spot to start association is 30m from AP, `Cellular-Agent` will use it to guide `Smart-Client` through a timeout value which indicates when `Smart-Client` should turn on its WiFi interface for association and then start downloading.

By setting up `Cellular-Agent` and `WiFi-Agent`, we conducted the experiments 10 times for each direction, to the east and to the west. We installed `Smart-Client` on our prototyped Nokia N900 smartphone. We plot the results in Figure 4.15 and 4.16.

Regarding the content prefetching on WiFi AP, the prefetched volume shows the amount of data `WiFi-Agent` fetched from the Internet for the duration once it received the instruction from `Cellular-Agent` till the point `Smart-Client` walked away from the WiFi coverage (i.e., disconnected from the WiFi). If `Smart-Client` finishes the downloading of all prefetched data when the smartphone is still connected to the AP, `Smart-Client` will download some dummy files to measure the actual offloaded volume.

As we can see in Figure 4.15 and 4.16, for around 85% of the runs, we can fully utilize the WiFi link (i.e., the amount of downloaded data is smaller than that of prefetched data with an average gap around 4.8 MB). The average offloading volumes in our experiments are 27.168 MB and 22.861 MB, for walking to the east and walking to the west, respectively.

Regarding the location where `Smart-Client` starts the WiFi association following the predicted timeout value instructed by `Cellular-Agent`, we plot the results of 10 runs for each direction in Figure 4.17. On average, when walking to the east, `Smart-Client` starts to associate with the AP when it was around 33.3±3.56 meters away from the AP. The average distance was 29.9±4.65 meters when walking to the west.

To understand the overhead of WiFi-based offloading, we also plot the number of WiFi association attempts to connect to the AP in Figure 4.18 for the 10 runs of each direction. The average association attempts with standard deviation were 4.4±2.01 for walking to the east, and 4.6±0.84 for walking to the west. We note that such high number of associations experienced in the live environment can lead to extra energy cost. We consider such cost $E_{asso}$ as part of the overhead $E_{oo}$ in the energy saving evaluation.

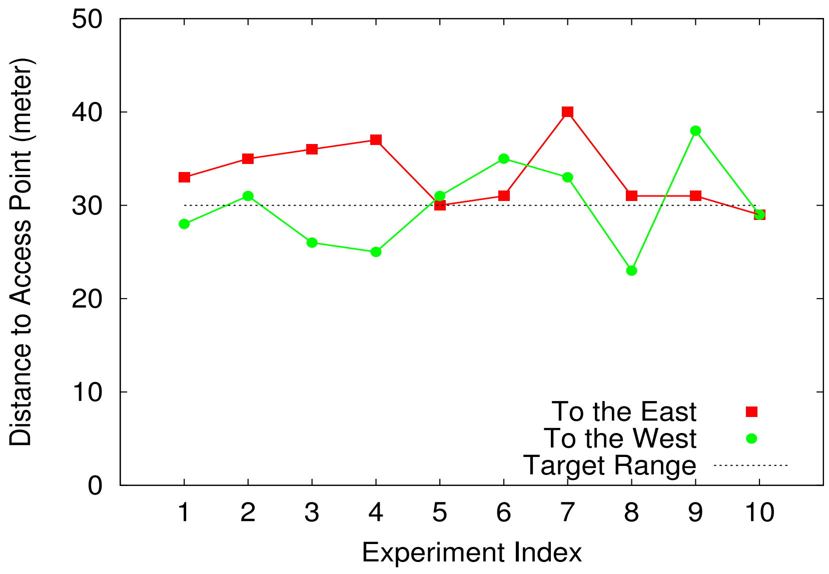


Figure 4.17: Predicted association [16].

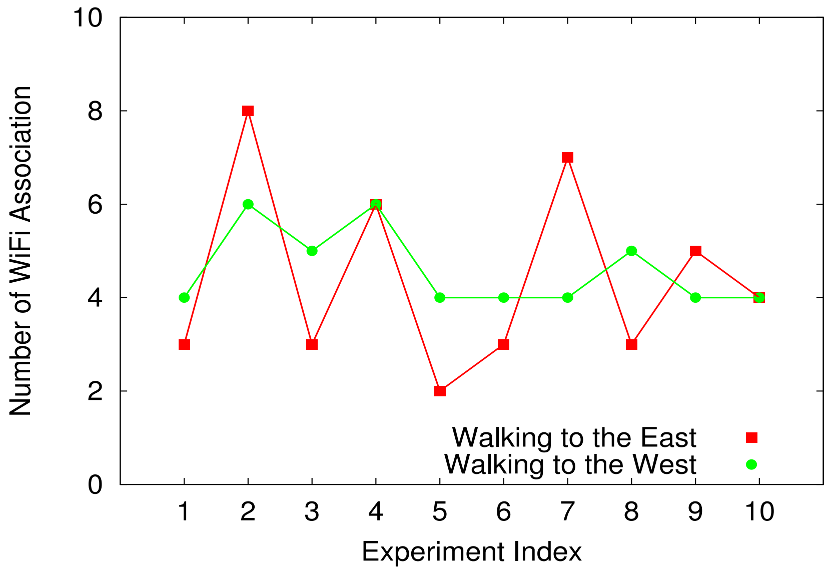


Figure 4.18: Number of association attempts.

Table 4.7: Average measured power (Watt) and energy consumption (Joule) on N900.

| Device | $P_{3G}$ | $P_W$ | $E_T$ |
|---|---|---|---|
| N900 | 1.10±0.017 Watts | 0.645±0.023 Watts | ∼5.4 Joules |

***Energy Savings***

As we focus on downlink streaming traffic in MADNet, we summarize our measurements in Table 4.7, including the transmission power for both 3G and WiFi ($P_{3G}$ and $P_W$), using Monsoon Power Monitor with 5 KHz sampling rate. For N900, the head and tail energy $E_T$ of 3G is ∼5.4 Joules, which is used in our evaluation.

We emphasize that another potential benefit of using 3G as a control channel is that smartphones can scan only the channel of the specified WiFi AP and avoid complete scanning of all possible channels. For the real deployment of MADNet, we can get the power values using either existing online estimation tools (e.g., PowerTutor [153]) or offline profiles built for different types of smartphones.

We estimate the energy saving of offloading mobile data traffic to the deployed outdoor AP using the results in Table 4.6 and 4.7, in combination with the live traces from our outdoor experiments. We present the energy savings for the walking experiments in Table 4.8, where the first row of the results is for walking to the east scenario and the second row is the scenario of walking to the west. $C_W$, $B_{3G}$ and $B_W$ represent the average offloading capacity, average throughput of 3G and WiFi networks, respectively, of 10 test cases.

$E_{3G}$ and $E_W$ are the energy consumption of data transfer through 3G and WiFi networks, respectively. $E_{3G}$ is calculated by using $E_{3G} = P_{3G} \times C_W/B_{3G}+E_T$. For $E_W$, we use the measured values of outdoor experiments to calculate it, by using $E_{W\_i} = P_W \times C_{W\_i}/B_{W\_i}$. We report the mean value of 10 test cases (i.e., i = 1, ... ,10) for walking towards east and towards west, respectively.

We turn on the WiFi radio twice during the offloading, first for WiFi localization and the second time for WiFi association. During WiFi fingerprinting-based localization, `Smart-Client` scans neighboring APs three times to collect WiFi beacons. $n$ is the number of associations to connect to the WiFi AP as shown in Figure 4.18. Thus the value of $E_{oo}$ is set as

$$E_{oo} = 2 \times E_{W\_on} + 2 \times E_{W\_off} + 3 \times E_{scan} + n \times E_{asso}.$$

Table 4.8: Measured average offloading capacity, average throughput of 3G and WiFi networks, average energy consumption of overhead, 3G and WiFi, and estimated energy saving [16].

| | $C_W$ | $B_{3G}$ | $B_W$ | $E_{oo}$ | $E_{3G}$ | $E_W$ | Saving |
|---|---|---|---|---|---|---|---|
| | MB | Mbps | Mbps | Joule | Joule | Joule | % |
| East | 27.2 | 0.8 | 3.5 | 3.4 | 304.6 | 39.3 | **85.98** |
| West | 22.9 | 0.8 | 3.2 | 3.5 | 257.3 | 38.9 | **83.52** |

Table 4.9: Estimated energy saving for different 3G throughput [16].

| | 0.5 Mbps | 1.0 Mbps | 1.5 Mbps | 2.0 Mbps | 3.0 Mbps | 5.0 Mbps |
|---|---|---|---|---|---|---|
| East | 90.7 % | 81.7 % | 72.9 % | 64.2 % | 47.5 % | 16.3 % |
| West | 89.1 % | 78.5 % | 68.2 % | 58.2 % | 38.9 % | 3.4 % |

The $E_{oo}$ in Table 4.8 represents the average value of 10 test cases for walking towards east and towards west, respectively. We calculate the energy saving as

$$Saving = 1 - (E_W + E_{oo})/E_{3G}$$

The results presented in Table 4.8 suggest that MADNet can achieve more than 80% energy saving on smartphones when offloading mobile traffic to WiFi networks, benefiting from the collaborative MADNet architecture.

We also note that we did not consider the energy consumption of control information exchange between `Smart-Client` and `Cellular-Agent` due to the small amount of data exchanged (usually less than 1KB). As these information is transferred in parallel with the music streaming, the impact of head and tail energy is kept to a minimum. Moreover, we use static IP address configuration enabled by the control channel of MADNet and thus avoid the energy consumption for getting an IP address through DHCP. Therefore, the energy saving reported in Table 4.8 shows the upper bound that can be achieved by MADNet.

To gain more insights, we also present the estimated energy savings in Table 4.9 for scenarios where the 3G throughput is set virtually to be 0.5, 1.0, 1.5, 2.0, 3.0, and 5.0 Mbps. The key observation is that when 3G throughput increases, the gain of energy consumption by offloading mobile data to WiFi will decrease.

In brief, mobile traffic offloading for moving smartphones is challenging, including positioning and prediction for the mobility and target WiFi AP. We acknowledge that our experimental evaluation relies on a few assumptions such as pre-collected traces of location coordinates, knowledge of wireless access throughput, and WiFi fingerprint radio-maps. These assumptions simplify our test setup and let us focus on evaluating the performance of our system through outdoor experiments. As the related areas have been investigated [7, 77, 124], the assumptions we made for our experiments rely on their results and suggestions. Nevertheless, the evaluation results show the energy saving and efficacy of our proposal under such assumptions, which can be regarded as the performance upper bound for MADNet in a live environment.

## 4.5 Discussion

In this section we discuss open issues in WiFi-based traffic offloading and compare MADNet with related work.

#### ***Incentives for WiFi Offloading and Deployment***

As social participation is becoming an enabling factor for more and more mobile applications, how to integrate an effective incentive scheme into MADNet to encourage active participation is a challenging problem. Cellular operators may offer reduced subscription fees to users who are willing to share their home APs with others, and thus increase the availability of WiFi access. There may also be issues related to the pricing and billing models for traffic offloading between different cellular operators and WiFi providers. For example, in countries like Canada where usage-based billing has been introduced for Internet access, people cannot share their home APs with others for free and mobile users may need to pay for the traffic offloaded to WiFi networks.

Due to the chaotic nature of AP deployment and wireless link characteristics, there are coverage holes for the current WiFi network deployment in metropolitan areas. As indicated by a recent study [12], one of the key goals is to minimize the number of deployed WiFi APs to reduce various costs, by considering the non-uniform propagation of WiFi signals and the application usage patterns of mobile users. To this end, realistic models of user mobility and content requests are also important factors in efficient deployment.

One visible trend to increase the number of APs for traffic offloading in metropolitan areas comes from operators who deploy their own WiFi networks [11] [12] [13]. The collaborative design of MADNet and our proposed optimization schemes can help bridge the management gap between cellular and WiFi networks

#### ***Distributed Content Caching***

Usually, WiFi service providers prefer to over-subscribe the backhaul connections from the APs to Internet and thus these backhaul links become the communication bottleneck [272]. In MADNet, we use the limited storage space (e.g., several GB through mounted USB drives) on WiFi APs to cache data locally. The challenge here is to understand the content access patterns from mobile users, without which it is hard to decide the right locations to cache the right content. Similar to data prefetching, caching data on local APs can also fully utilize the bandwidth of WiFi links.

#### ***Security and Privacy***

For security and privacy, when MADNet decides to offload mobile traffic to WiFi APs, cellular networks can pass the identity information of end users through the MADNet controlled channel to WiFi networks and thus enable various security models to prevent illegal uses of these APs. Meanwhile, we may require mobile users to tunnel their packets to one of their trusted points and handoff the access control responsibility to that endpoint [265]. Thus, all the offloaded traffic will go through that trusted point and can be identified if the mobile users illegally download music or video. For example, SWISH [266] has utilized a similar technology to secure the shared WiFi networks and to protect the privacy of mobile users. We note that mobile traffic offloading should be transparent to content/cloud service providers and MADNet will not reveal the end user's location to them.

#### ***Support for Internet Host Mobility and Handoffs***

Mobile IP is a standard communication protocol that allows a mobile device to change its Internet attachment point while maintaining a permanent IP address. To improve the scalability of Mobile IP, Cáceres and Padmanabhan [267] propose a hierarchical mobility management scheme by exploiting the geographic locality in user mobility patterns. In their solution, the frequent movement within subnetworks is handled by the lowest level of the hierarchy and the movement between subnetworks relies on the

[11] Cable WiFi Internet Access: `http://www.cablewifi.com/`
[12] Verizon WiFi: `https://www.verizon.com/home/wifi-wireless-internet-service/`
[13] AT&T Wi-Fi Internet Service: `http://about.att.com/mediakit/wifi`

original Mobile IP, and thus it can reduce handoff latency and the Internet traffic load. Motivated by the observation that IP multicasting and Mobile IP share similar issues of location independent addressing and packet forwarding and routing, Mysore and Bharghavan [268] propose a multicast-based architecture for Internet host mobility. In order to support seamless mobility, K. Pahlavan, M. Ylianttila, et. al. have investigated handoff mechanisms in multi-access mobile environment [63, 64, 65, 66, 67].

Besides supporting host mobility with network-layer solutions, there are also several protocols at higher layers to achieve the same goal. Heterogeneous wireless networks, also called wireless overlay networks [55], refer to the integration of multi-technology wireless networks (e.g., cellular, WiFi, Bluetooth, WiMax and ZigBee) with different access methods. Snoeren and Balakrishnan [269] design an end-to-end approach for host mobility, which requires no changes to the IP substrate. The proposed solution modifies the transport-layer protocols at the end hosts, takes advantage of the secure dynamic updates in DNS to track host location, and designs a new TCP option to migrate established TCP connections when the IP address of an end host changes.

Zhuang et al. [270] propose the Robust Overlay Architecture for Mobility (ROAM), which uses Internet Indirection Infrastructure ($i^3$ [271]) to provide mobility for end hosts. In ROAM, each data packet is bound with an identifier, which defines an indirection point in $i^3$, and a receiver can obtain the packet based on its identifier. The prototype of ROAM is a user-level system that requires no modification of the TCP/IP stack and has the following nice properties: efficient routing and handoff, fault tolerance, location privacy, and simultaneous mobility.

There are several projects in this area [272, 81], following one of the seminal works by Stemm and Katz [55], which proposes mobile-IP-based vertical handoffs that occur between different access techniques. Buddhikot et al. implement IOTA [272] for integrating two access technologies (CDMA2000 and WiFi), which has been operational on a testbed for years. Chakravorty et al. [81] present a measurement study of inter-networking mobility between GPRS and 802.11b hotspots with a focus on its impact on active TCP flows. Their analysis shows that the disparity in RTT and bandwidth of GPRS and WiFi networks can aggravate the performance of vertical handovers.

Due to the strict QoS requirements of real-time applications, MADNet only uses WiFi networks when possible to augment cellular networks and will not switch back and forth between these two networks.

### *Leveraging Multiple Radio Interfaces on Mobile Devices*

Nowadays smartphones are typically equipped with multiple radio interfaces, including Bluetooth, WiFi and 3G cellular, and there are several existing systems leveraging these interfaces for energy efficiency and better throughput performance. For example, CoolSpots [182] explores policies that enable a mobile device to automatically switch between WiFi and Bluetooth interfaces by considering their different transmission ranges, and thus reduces the energy consumption of wireless communication. MAR [104] is a multi-homed mobile access router that exploits multiple channel access technologies provided by different service providers. The MAR implementation demonstrates the benefits of aggregating link capacity from three cellular operators for different applications.

Ra et al. [166] study the energy-delay tradeoff for smartphone applications and design an online network interface (e.g., 3G, EDGE, or WiFi) selection algorithm. Context-for-Wireless [180] explores context information, including cellular network conditions, device motion, time and history, to estimate network conditions, and then chooses wireless interfaces for data transfer based on these estimations. Super-aggregation [181] is a strategy solution that aggregates the link capacity of multiple interfaces on mobile devices and the achieved aggregation throughput is more than the sum of the parts. It introduces three principles, named selective offloading, proxying and mirroring, to exploit the large degree of bandwidth heterogeneity of these interfaces. With the goal of prolonging battery life, Armstrong et al. [273] propose to select between GPRS and 802.11 interfaces based on message size, for dynamic content updates on mobile devices. With the focus of minimizing energy consumption, the algorithm implicitly delays data packets until the underlying link quality is good. Intentional Networking [274] leverages a declarative label of various applications for transmissions over multiple wireless networks and thus can match network traffic to the most suitable network interface.

Compared with previous work, MADNet uses 3G networks to provide primary data channels and to facilitate offloading procedures. At the same time, MADNet employs WiFi interfaces to prefetch predictable data and thus improves the energy efficiency of data transfer.

### *Traffic Offloading Design*

Among several existing schemes for cellular traffic offloading, Femtocells as an extension of the macrocells of cellular networks were originally proposed to offer better indoor services. When indoor users switch from macrocells to femtocells, femtocells can potentially offload cellular data traffic. However, the femtocell signal may interfere with nearby macrocell transmissions, since they work on the same spectrum as macrocells [263]. Recently, how to offload cellular traffic through mobile-to-mobile opportunistic communications is also investigated as a complementary offloading solution [93].

WiFi is another attractive technology for cellular traffic offloading. For example, Korhonen et al. [26] analyze the latest trend of network controlled offloading and compare industrial standardization solutions.

There are several WiFi-based cellular traffic offloading schemes proposed by the industry. For example, the uAxes offloading solution from Notava[14] proposes the concept of WiFi capacity marketplace and designs a global trading system to buy and sell surplus WiFi capacity online. The MDO gateway from Stoke[15] can alter the traffic path of Internet connections, bypassing the cellular core networks, and forward mobile data traffic directly to the nearest Internet access point.

Hou et al. [8] propose a transport layer protocol to offload 3G data traffic to WiFi hotspots for vehicular access networks. Ristanovic et al. [11] propose an algorithm, called HotZones, to offload delay-tolerant cellular content to WiFi APs and evaluate the performance through trace-driven simulations. Wiffler [5] aims at maximizing the amount of 3G traffic offloaded to WiFi networks for PCs on vehicular networks. In contrast to Wiffler, MADNet targets smartphones which are much more challenging than PCs (as discussed in Section 4.2.1). MADNet also advances the state-of-the-art by taking the energy consumption of smartphones into account when making offloading decisions. In addition, Wiffler uses only open WiFi APs while MADNet also considers the support from cellular operators, WiFi providers, and end users to increase the offloading opportunities.

Using a trace-driven simulation, Lee et al. [6] show that WiFi networks can offload around 65% of mobile data traffic for the traces collected from about 100 iPhone users. The simulator assumes that data traffic can be offloaded to WiFi networks *whenever a user connects* a smartphone to a WiFi AP in the traces, thus the offloading decision is actually made by these users. Compared to their work, our proposed offloading solution is transparent to users. Furthermore, we implement a prototype on Nokia N900 smartphones and evaluate its performance in live environment.

As pointed out previously, performance evaluation based on desktop/laptop computers does not necessarily reflect that experienced by mobile phones [264]. A recent measurement study examining the relationship of WiFi versus 3G

[14] `http://www.notava.com/`
[15] `http://www.stoke.com/`

usage also explored the antenna limitation of smartphones [15]. Although the target is different from ours, their findings confirm that a dedicated offloading scheme for smartphones is necessary.

#### *Limitations*

We note that the limitations of our work include that we have evaluated only a few scenarios, two experimental environments in Maryland and Helsinki, a few WiFi APs and smartphones (including one N900 for prototyping), covering only downlink traffic. In addition, several assumptions were made for evaluating the performance and the impact of prediction errors on MADNet. Regarding the energy-aware offloading algorithm, 3G and WiFi access capacity may vary with location and over time. Our approach requires such context information as a priori to achieve accurate predictions, which may introduce non-negligible overhead such as to profile the targeted area. Rather than discussing general applications for smartphone, we mainly investigate streaming applications that contribute a large amount of data to the mobile traffic [1] where the pre-fetching technique is reasonable by using the available playlists. However, some of the issues such as offloading efficiency and scalability that are not covered in our work have been examined in previous work, although for different purposes [5, 6, 15, 85, 86, 101, 261, 275]. Therefore, we choose to focus on the unsolved issues of energy saving for smartphones in WiFi-based offloading and leave the other as our future work.

## 4.6 Summary

We propose MADNet, a collaborative WiFi-based mobile data offloading architecture that aims at improving energy efficiency for smartphones. It utilizes the proposed energy-aware offloading algorithm to assist the offloading decision for smartphones, which enables them to select the most energy efficient WiFi APs to offload mobile traffic.

The design choices are motivated by the measurement study of war-driving and war-walking and experimental results of WiFi-based Internet access for moving smartphones. By considering the battery constraints of smartphones, we design the first energy-aware offloading decision algorithm with the goal of improving energy efficiency for smartphones.

We evaluate the performance of our system using our prototype implementation on Nokia N900 smartphones. We demonstrate that by leveraging a deployed outdoor WiFi access point, our solution is able to achieve more than 80% of energy saving. We also show that the potential energy saving of offloading depends on the throughput of 3G and WiFi networks, and also the amount of data we can offload.

In brief, we make the following contributions:

- We investigate and compare the performance of WiFi Internet access for both netbooks and smartphones in metropolitan areas. Our results indicate that the throughput of WiFi-based Internet access for smartphones is lower than PCs, due to their restricted antennas for outdoor communication. The observations offer valuable insights for future design and inspire us to develop optimization schemes to support WiFi-based offloading for smartphones.
- We propose MADNet energy-aware offloading architecture for smartphones, which explicitly considers the potential contributions of cellular operators, WiFi providers, and end users. We design an efficient and deployable *energy-aware* algorithm to assist the offloading decision. Our algorithm, integrated with the collaborative MADNet architecture, is able to tolerate minor errors of input parameters such as user location, mobility prediction, and estimation of system offloading capacity. We also utilize content prefetching to compensate for the impact of hardware limitations on smartphones.
- Our prototype implementation confirms the feasibility of MADNet to be deployable in existing environments. By enabling smartphones to select the most energy efficient WiFi AP for offloading, our field experiments further suggest that MADNet can achieve a notable 80% of energy saving. To encourage collaboration and future development, we release the software implementation under open-source licenses [16].

Compared with NAO, we bring WiFi operators into the collaboration through the MADNet collaborative offloading framework so that both network side and user side can contribute and benefit from their joint efforts. However, our research oriented design suffers from its limitation such as lack of openness and extensibility. The insight from MADNet sheds lights on our future work towards an open, extensible and deployable design.

[16] MADNet: `http://www.cs.helsinki.fi/u/yding/src.html`

# Chapter 5

# Software-Defined Collaborative Traffic Offloading

This chapter presents our software-defined platform, SoftOffload, for achieving collaborative traffic offloading. Section 4.1 describes the motivation that drives us to design an extensible and deployable offloading platform. Section 4.2 presents our measurement studies that explore hidden issues in WiFi offloading. The observations inspire us to develop optimization schemes for SoftOffload. Section 4.3 introduces the system design of SoftOffload. Section 4.4 illustrates the system implementation and experimental evaluation. Section 4.5 summarizes the chapter.

## 5.1 Motivation

As discussed in previous chapters, mobile traffic is expected to surge and licensed spectrum is becoming increasingly scarce. Using WiFi for mobile traffic offloading (WiFi offloading) is gaining importance for relieving congestion in mobile networks. In fact, with the exception of smartphones, recent studies have shown that WiFi is the predominant access technology for most mobile devices, such as laptops, netbooks, tablets and eReaders, to connect to the Internet [278, 279].

Along with public WiFi deployed by public sectors, carrier-class WiFi deployed by MNOs is an attractive complement to mobile small cells. Vendors such as Ericsson have launched multi-radio small cell products that can support WiFi [1]. Studies also show that carrier-class WiFi deployments allow MNOs to save 30-40% on network CAPEX while gaining wireless access capacity in typical indoor deployments [280]. In addition, WiFi offloading

[1] `http://www.ericsson.com/us/ourportfolio/telecom-operators/rbs-6402`

also keeps the need for expensive and scarce 4G/LTE licensed spectrum low while enabling MNOs to satisfy the surge in demand for wireless access capacity.

Regarding one of the major contributors to mobile data traffic, video streaming, the advancement in video encoding and playback (e.g., 3D, high quality encoding, 4K resolution) will further push up the capacity requirements. This inevitable demand puts WiFi offloading into a compelling position to share the burden of traffic growth on mobile networks.

However, given all the potential benefits WiFi offloading can deliver, there are issues remaining to address:

- First, we need to understand the impact of WiFi offloading on end users. Previous work [16] has shown that using 'weak WiFi' access for offloading can negatively affect transmission performance and energy consumption on smartphones (discussed in Chapter 4). The term 'weak WiFi' refers to two aspects: weak signal strength and weak access capacity. For example, if the signal strength is weak, wireless losses and retransmission can prolong the transmission and thereby lead to high energy consumption. Regarding access capacity, if a WiFi access point (AP) is congested (e.g., used simultaneously by many other devices), even though the user-perceived signal strength is good, using this AP for offloading will result in poor performance and extra energy cost. A quantitative analysis on this topic regarding performance and energy consumption is still missing.

- Second, we need an efficient and collaborative platform that is suitable for dynamic changes in wireless access environments. For WiFi offloading in particular, we need to address how to efficiently collect essential information from different sources (e.g., cellular and WiFi networks) and synthesize these information to assist traffic offloading.

- Third, since communication technologies and mobile industry are advancing at a fast pace, a platform for WiFi offloading needs to be flexible and extensible to adopt new optimization schemes and hence meet the emerging requirements. For instance, such solution should not only cover the scenario of carrier-class WiFi integrated with cellular infrastructure, it can also be extended to support cloud service providers to offer WiFi services through a mobile virtual network operator (MVNO) channel.

A key challenge in WiFi offloading is that valuable contextual information is often distributed across network entities and end users. This unbalanced distribution indicates that the contextual information possessed by

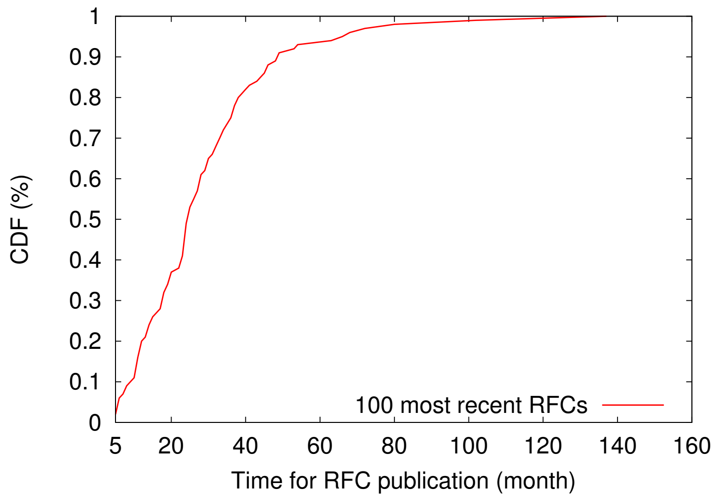


Figure 5.1: Time required for publishing RFC [17].

each side along is insufficient to guide the offloading decision. On the network side, the traffic load and condition of managed entities (e.g., router, APs) can be acquired to assist traffic offloading. For example, operators can decide to trigger traffic offloading when a cellular base station is overloaded by several active users. However, today it is difficult, if not impossible, for operators to capture the instant local context perceived by those mobile users (e.g., signal strength of nearby WiFi APs). On the other hand, mobile users can acquire local context by using WiFi scanning and embedded sensors on their devices, but they are hardly aware of the conditions on the network side (e.g., congestions on an aggregated router) before connecting to the access network. A typical example is that, even if a WiFi AP is relatively close to the user, showing good signal strength, this AP can be heavily loaded by data traffic generated by other devices. If a user chooses to offload traffic using this congested AP (with good signal strength though), the outcome could be suboptimal.

Regarding another challenge, solution deployment, pushing a proposal to become a standard helps improve the deployability, especially compared with research-oriented solutions with limited scope and poor visibility toward mass markets. However, as discussed recently [17], standardization process can be time-consuming. Figure 5.1 illustrates the time requirement for publishing IETF Request for Comments (RFC) as an example. From the 100 recent RFCs shown in the figure, ranging from RFC 6931 to RFC

7038, around 50% of them demanded more than two years of devotion. The longest one goes even up to 137 months (i.e., 11+ years). In addition, there are also various invisible elements in the initial phase before the standardization procedure begins, including the time for selecting or setting up working groups, initiating the topics in the community, and day-to-day time commitment to keep up with the relevant discussions in the working groups.

Because the surge of mobile traffic necessitates the capacity provision of mobile networks to catch up, we are in urgent need for an agile, collaborative and cost-effect approach to orchestrate and maximize the utilization of resources in today's fast evolving multi-access environment comprising different wireless technologies. Software-defined networking (SDN) offers a centralized control model, a flow-based paradigm and programmability that is ideally suited for highly scalable mobile and wireless networks. For mobile networks, SDN is considered to bring tremendous operational expenditure (OPEX) improvements through the scalability of network control functions, the flexibility to adjust network resource allocation, and steer traffic dynamically in response to changing traffic patterns and network conditions. Since SDN delivers a fine-grained control of network and services through its abstraction of the underlying hardware, it meets the urgent need from the mobile networks to simultaneously operate over multiple wireless technologies (e.g., 4G and WiFi) in order to accommodate the radical growth of data traffic.

Since mobile traffic offloading involves cellular operators, alternative wireless access providers, and mobile users, it is necessary to establish collaboration among all the parties. For WiFi offloading, we should consider the restriction of mobile devices and the context of the mobile environment, such as user mobility, network load, user preference, and network policies. In this chapter, we investigate the following research questions:

- *What context information is important for WiFi-based traffic offloading? How do we obtain and utilize these context information efficiently to assist traffic offloading?*
- *How do we transfer a static and closed design to a dynamic and open one and thereby enhance the deployability and extensibility of our proposal?*
- *Can we apply the software-defined networking (SDN) design to address the challenge? How do we integrate the SDN paradigm into mobile traffic offloading?*

- *What are the benefits and overhead for an SDN-based offloading solution?*

We advocate that the way forward is to integrate SDN design into WiFi offloading and fully utilize the features of SDN in terms of programmability and openness for improving extensibility, interoperability and deployability. We therefore propose an open-source SDN-based platform, SoftOffload, to achieve optimal traffic offloading by considering different contextual factors.

## 5.2 Measurement Study

This section presents our measurement study that aims to answer the following questions:

- *Is the performance of latest WiFi technology good enough to support mobile traffic offloading, especially when compared with 3GPP 4G/LTE technologies?*
- *What types of contextual information are important in WiFi-based traffic offloading?*
- *What is the impact of such contextual information? Can we efficiently obtain and utilize these information to assist WiFi-based traffic offloading?*

### 5.2.1 WiFi offloading performance and energy consumption

Our measurement study focuses on WiFi-based traffic offloading, covering the performance and energy consumption in the cellular and WiFi networks, the impact of signal level on WiFi offloading, the competing traffic scenarios, and a revisit of the WiFi access availability.

Figure 5.2 shows the equipment used in our measurements. For power measurement, we used the Monsoon Power Monitor [2] to measure the energy consumption related to traffic offloading. The mobile devices include smartphones and tablets. We summarize the specification of our testing devices in Table 5.1, covering the year of release, model, CPU, memory size, screen size, battery capacity, network connectivity, and operating system (OS) installed at measurement time. Since we use Monsoon to conduct power measurement, we carefully select the smartphone models that are battery-removable, as illustrated in Figure 5.2 where we attached power

[2] `http://www.msoon.com/LabEquipment/PowerMonitor/`

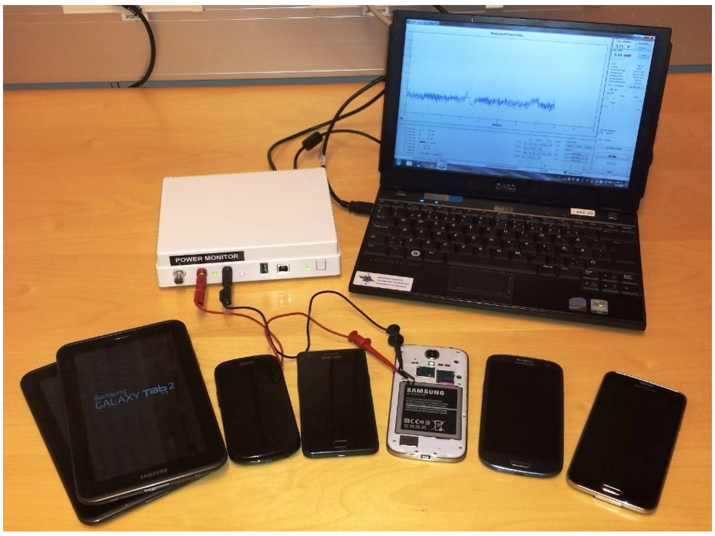


Figure 5.2: Equipment for measurement study.

Table 5.1: Specification of mobile devices used in measurement.

| | Nexus S | Galaxy S2 | Galaxy S3 | Galaxy S4 | Galaxy S5 | Galaxy Tab 2 |
|---|---|---|---|---|---|---|
| Release | 12.2010 | 02.2011 | 05.2012 | 03.2013 | 02.2014 | 02.2012 |
| Model | Nexus S | GT-I9100 | GT-I9300 | GT-I9505 | SM-G900F | GT-P3100 |
| CPU | 1-Core, 1GHz | 2-Core, 1.2GHz | 4-Core, 1.4GHz | 4-Core, 1.6GHz | 4-Core, 2.5GHz | 2-Core, 1GHz |
| RAM | 512 MB | 1 GB | 1 GB | 2 GB | 2 GB | 1 GB |
| Screen | 4.0 inches | 4.3 inches | 4.8 inches | 5.0 inches | 5.1 inches | 7.0 inches |
| Battery | 1500 mAh | 1650 mAh | 2100 mAh | 2600 mAh | 2800 mAh | 4000 mAh |
| Network | HSPA, 802.11 b/g/n | HSPA+, 802.11 a/b/g/n | HSPA+, 802.11 a/b/g/n | 4G/LTE, HSPA+, 802.11ac | 4G/LTE, HSPA+, 802.11ac | HSPA+, 802.11 b/g/n |
| OS | Android v4.1.2 | Android v4.1.2 | Android v4.3 | Android v4.4.2 | Android v5.0 | Android v4.1.2 |

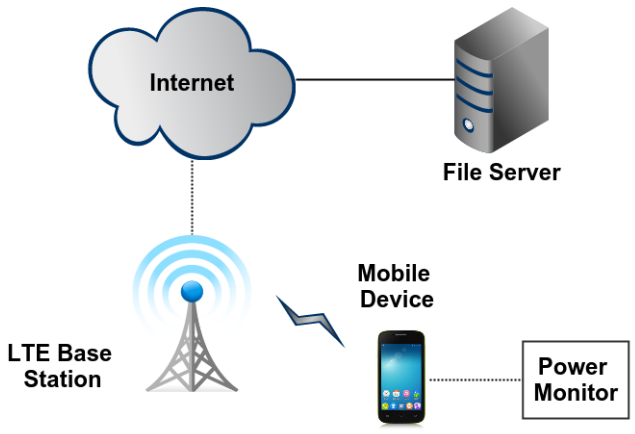


Figure 5.3: Cellular test setup.

supply cords to the back of a Galaxy S4 smartphone with a removable battery.

***Performance and Energy Consumption in Cellular and WiFi***

Regarding the benefits of WiFi-based offloading, one typical question is that given the fast development of cellular access technologies, such as Long Term Evolution (LTE) [3], will the speed of cellular access catch up or even surpass that of WiFi, thereby making the WiFi-based offloading less appealing for end users? Compared with the downlink rate of 54 Mbps offered by 802.11g, the claimed peak rate of 300 Mbps supported by LTE does make the question relevant and worth a thorough investigation, especially in operational networks.

First, we conducted a series of measurements to investigate the transmission rate and energy consumption in an operational LTE network in Helsinki Finland. Figure 5.3 illustrates the setup of our cellular measurement. Since downlink traffic such as video streaming is a key contributor to mobile data traffic, this measurement focuses on the downlink speed and the corresponding energy consumption.

To reveal the practical transmission rate perceived by modern handsets, we selected the LTE-capable models, Galaxy S4 and Galaxy S5, as the end devices. We developed a testing Android application for this measurement, which will execute a download task using HTTP to fetch a 15 MB file from

[3] `http://www.3gpp.org/specifications/releases/72-release-8`

Table 5.2: Throughput (Mbps) and energy consumption (Joule/MB) for downloading 15 MB in cellular networks.

| | Galaxy S4 | | Galaxy S5 | |
|---|---|---|---|---|
| | Throughput | Energy consumption | Throughput | Energy consumption |
| LTE | 46.025±5.71 | 0.33±0.03 | 42.131±0.786 | 0.254±0.013 |
| HSPA+ | 9.792±0.532 | 1.28±0.068 | 12.888±2.256 | 0.814±0.106 |

Table 5.3: Throughput (Mbps) and energy consumption (Joule/MB) for downloading 33 MB in WiFi networks.

| | Galaxy S4 | | Galaxy S5 | |
|---|---|---|---|---|
| | Throughput | Energy consumption | Throughput | Energy consumption |
| 802.11ac | 186.25±19.43 | 0.131±0.013 | 245.33±16.72 | 0.104±0.009 |
| 802.11n | 37.01±1.33 | 0.338±0.009 | 39.3±1.48 | 0.372±0.007 |

an HTTP server located in the Department of Computer Science, University of Helsinki. We chose an operator in Finland offering LTE subscription with the maximum downlink speed of 150 Mbps. For comprehensiveness, we also measured the performance of HSPA+ on the selected devices. We repeated each test five times and reported the mean values and standard deviations in Table 5.2. All cellular measurements were carried out in the same location at the laboratory during workday daytime. The received signal strength indicator (RSSI) values in LTE measurements were around -76 dBm. For HSPA+ cases, RSSIs were around -52 dBm [4].

Second, to investigate the question "*Is the performance of latest WiFi technology good enough to support mobile traffic offloading, especially when compared with 3GPP 4G/LTE technologies*", we measured WiFi performance covering 802.11ac and 802.11n. Figure 5.4 shows the testing layout. To reveal the wireless link speed in a controlled environment, we connected the file server to AP via 1 Gbps Ethernet without any middlebox in between. As shown in Figure 5.5, we used an ASUS RT-AC68U Dual-band gigabit router as the WiFi AP to test 802.11ac. We used a NETGEAR WNA1100 wireless USB adapter attached to laptop to measure the performance and energy consumption for 802.11n. We repeated each test five times and reported the mean values and standard deviations in Table 5.3.

[4] Under such RSSI values, our testing smartphones showed 5 signal bars (full), indicating good signal strength.

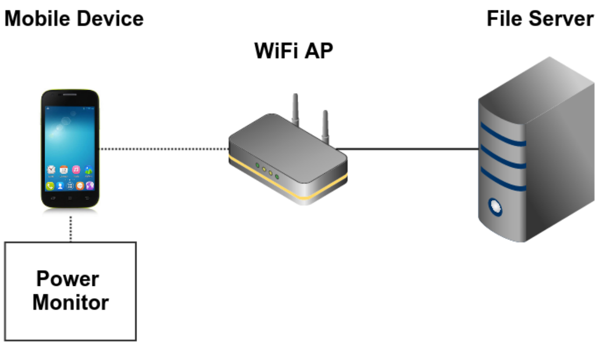


Figure 5.4: WiFi test setup.

The results in Table 5.2 and 5.3 suggest that the commercial cellular network we measured can already offer good performance. For LTE, the downlink speed can reach up to 46 Mbps, and HSPA+ can provide up to 12.8 Mbps for downlink traffic. On the other hand, the advance of WiFi technology is visible. While 802.11n in our controlled environment provides a downlink speed of 39 Mbps, the newer 802.11ac can even deliver a speed of 245 Mbps for downlink traffic.

Concerning the energy consumption, as highlighted in Figure 5.6 for modern models Galaxy S4 and S5, LTE is more efficient than HSPA+. Meanwhile, WiFi is in general an energy-efficient choice compared with cellular technologies. In particular for 802.11ac, it consumes less energy than LTE on the same smartphone model. In addition, 802.11ac consumes much less energy compared with HSPA+.

This observation indicates that WiFi is still a viable candidate for mobile traffic offloading owing to its advancement in terms of performance and energy efficiency. To further maximize the performance gain, we need to adopt optimization techniques such as content pre-fetching and caching (as discussed in Chapter 4) to avoid the backhaul capacity bottleneck. We note that our measurements cover limited scope (e.g., one cellular operator in Finland, a single cellular location, with limited repetitions). Similar studies have provided insights on the cellular and WiFi performance [6, 128, 129, 135, 145]. In this regard, our measurements cover the modern smartphones (e.g., Galaxy S4 and S5) and provide comparison insight to motivate our WiFi offloading design.

#### *Impact of Signal Level in WiFi Offloading*

One observation from our WiFi performance measurement is that the

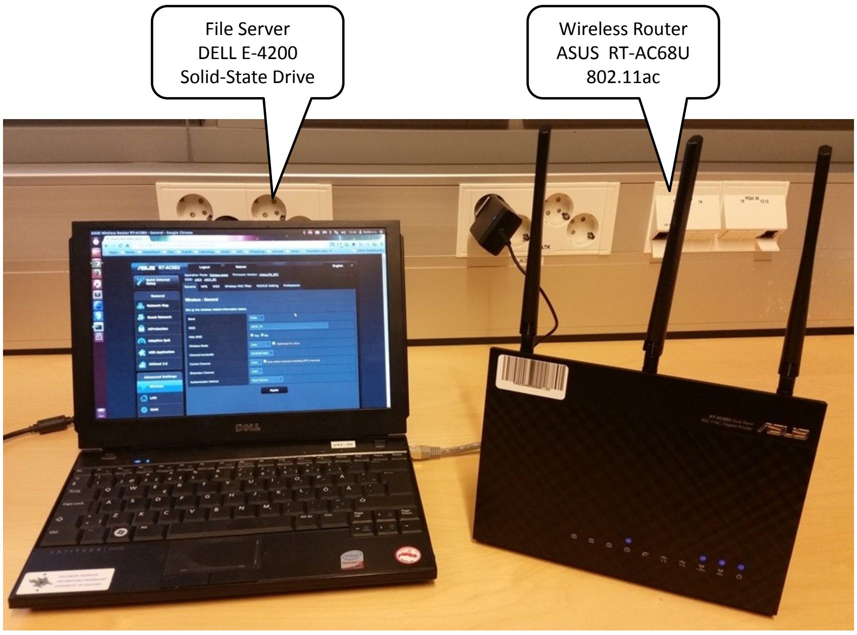


Figure 5.5: Equipment for evaluating 802.11ac.

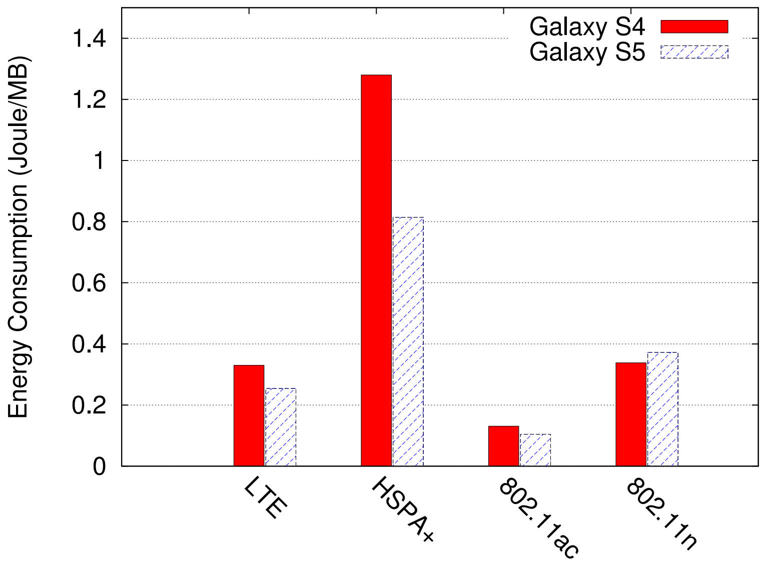


Figure 5.6: Energy consumption in cellular and WiFi.

correlation between transmission performance and WiFi signal strength (i.e., reflected by RSSI) is non-linear. We hence measured the throughput and energy consumption over 10 different RSSIs (ranging from -36 dBm to -77 dBm) in our laboratory. We repeated the measurement 5 times for downloading a 33 MB file at each RSSI and plotted the results for Galaxy S5 in Figure 5.7 and 5.8. As shown in Figure 5.7, when RSSI values are above -65 dBm, the impact of signal strength on throughput is relatively negligible. However, we observe a steep drop of throughput when RSSI approaches and goes below -70 dBm. The energy consumption results in Figure 5.8 show the same the pattern where the energy consumption radically increases when RSSI approaches and falls below -70 dBm. This finding matches the observation of our previous study that selecting a ‘weak’ WiFi to offload traffic can sometimes consume more energy than using cellular connection [16].

Given this observation, we further carried out a series of measurements using different models of smartphones to investigate the impact of signal strength in WiFi offloading. In order to measure energy consumption, we deliberately selected Nexus S, Galaxy S2, S3, S4, and S5. The specifications of these devices are shown in Table 5.1. We measured different signal levels by placing our WiFi AP at a set of pre-defined spots that are of different distance from the testing smartphones. We repeated each case 5 times and report our results for downloading a 33 MB file over 802.11n in Figure 5.9 and 5.10.

The measurement results under 802.11n match what we observed on S5 over 802.11ac. For all the smartphones we measured, the impact of signal level follows a non-linear correlation with transmission performance and energy consumption. When the signal level drops below -70 dBm, WiFi transmission performance will degrade and energy consumption will increase significantly.

This observation indicates that one of the key context for WiFi offloading is the signal strength perceived by end devices (i.e., RSSI values reported by WiFi scannings). To avoid using a ‘weak’ AP, we need to take into account the signal strength collected on the mobile devices and use these contextual information to guide end users in WiFi offloading.

### *Competing Traffic Scenarios*

In selecting a target WiFi AP for offloading, end users are often unaware of the conditions on the network side, such as the current work load on the access gateway and other competing users sharing the same WiFi AP. Since

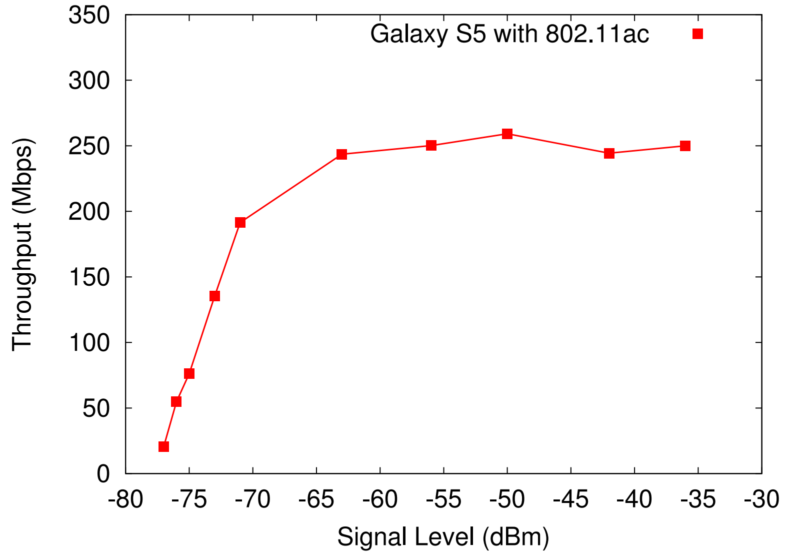


Figure 5.7: Throughput of S5 under different signal levels for downloading a 33 MB file.

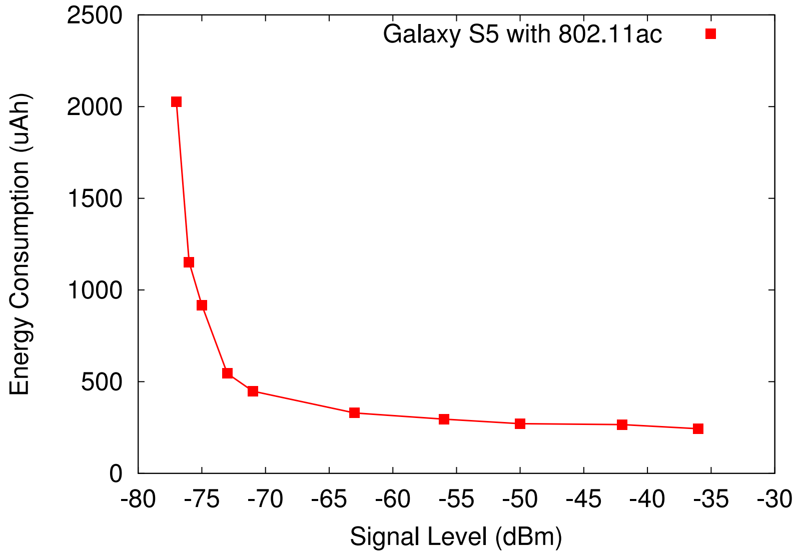


Figure 5.8: Energy consumption of S5 under different signal levels for downloading a 33 MB file.

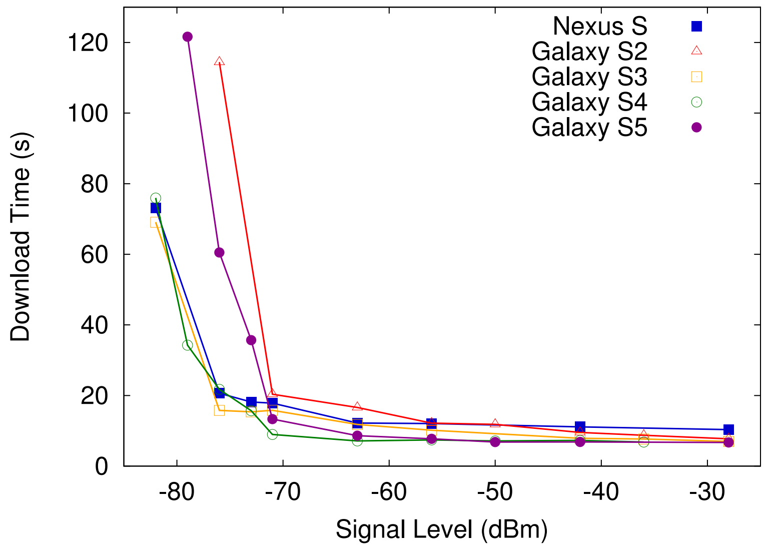


Figure 5.9: Download time under different signal levels for a 33 MB file.

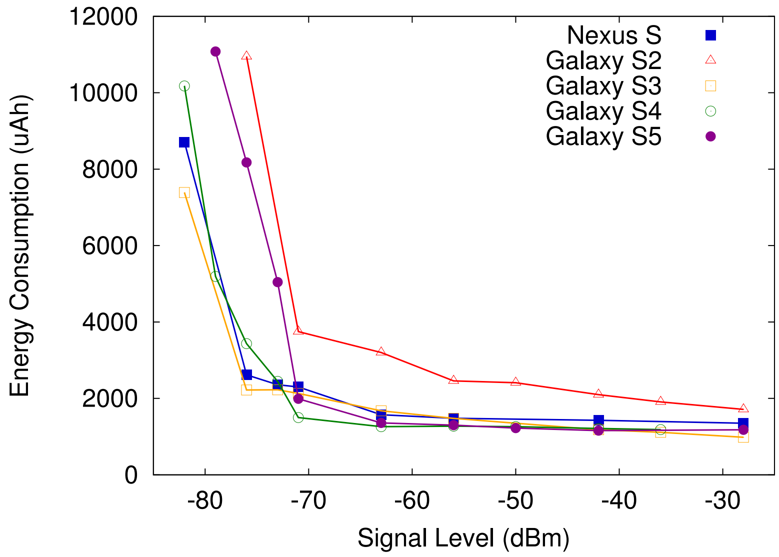


Figure 5.10: Energy consumption under different signal levels for downloading a 33 MB file.

these conditions may affect the overall performance of WiFi offloading, we need to understand their impact in competing scenarios when multiple active users are involved.

First, we study the case where multiple mobile devices are connected to the same AP. In this competing traffic scenario, we explore how much an individual user is affected in terms of transmission performance and energy consumption. In the experiment, we used the same layout as shown in Figure 5.4 with a file server running Apache HTTP server (version 2.0). For devices generating competing traffic, we selected two tablets of the same model (Galaxy Tab 2) as tablets are often used by mobile users for streaming services owing to their large screen size.

We measured the transmission time and energy consumption on the selected smartphones for downloading 33 MB file from the ASUS RT-AC68U router over 802.11n. The competing traffic were generated by tablets to download a 65 MB file through the same WiFi AP. The selected smartphones and the tablet were placed near each other with similar distance to the WiFi AP [5]. The smartphones and competing tablets started their downloading tasks at the same time. We consider cases when a smartphone is competing with one tablet and with two tablets.

We repeated each test case 5 times and report the results (mean value of 5 repetitions with standard deviation) for Galaxy S4 and S5 in Figure 5.11 and 5.12, on downloading time and energy consumption, respectively. We also conducted a series of measurements for legacy devices over 802.11g. We used the D-Link DI-524 AirPlus G router (running in 802.11g-only mode) as the WiFi AP and plotted the results for Nexus S, Galaxy S2, and S3 in Figure 5.13 and 5.14, for downloading time and energy consumption, respectively.

As shown in the figures, comparing with downloading alone without competition, all smartphones measured are affected by the competing traffic from other active devices sharing the same AP in terms of downloading time and energy consumption. This observation suggest that we need to consider the number of active devices that share the same AP as an important context in WiFi offloading.

Second, we study the scenario where multiple nearby devices are connected to different WiFi APs and compete over the same wireless channel. Figure 5.15 depicts our testing layout in the laboratory where two APs are placed 10 meters away from each other. To avoid measurement noise introduced by using different hardware models, we selected two tablets of the

[5] RSSI values were around -40 dBm

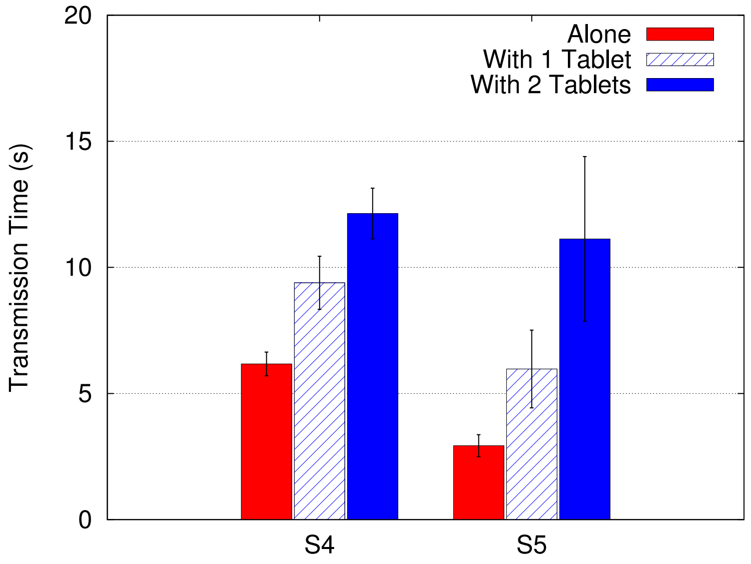


Figure 5.11: Download time of Galaxy S4 and S5 for a 33 MB file.

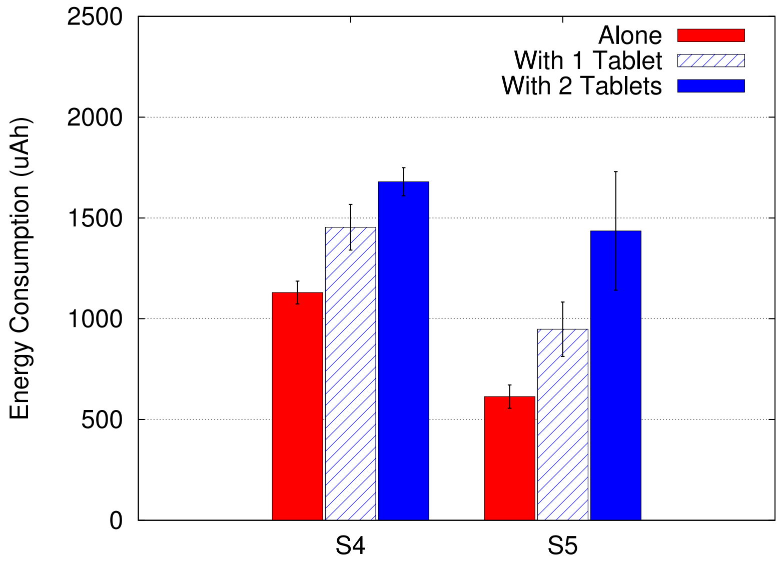


Figure 5.12: Energy consumption of Galaxy S4 and S5 for downloading 33 MB file.

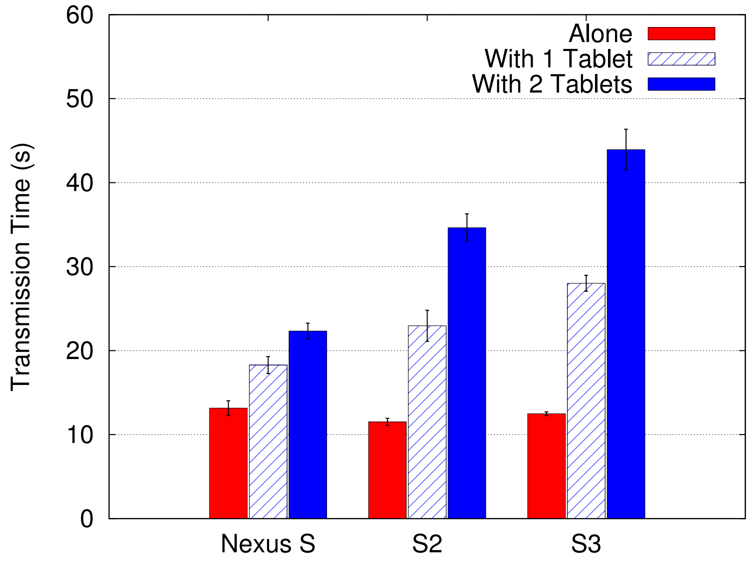


Figure 5.13: Download time for Nexus S, Galaxy S2 and S3 for a 33 MB file.

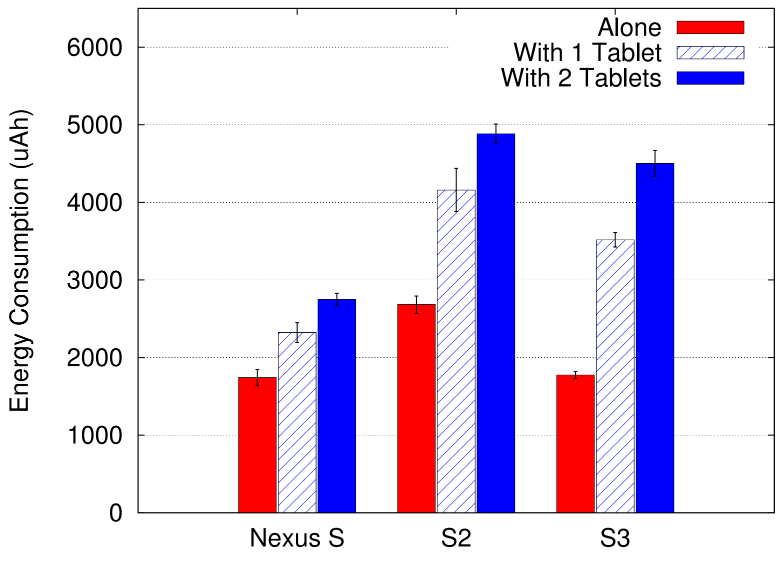


Figure 5.14: Energy consumption for Nexus S, Galaxy S2 and S3 for downloading 33 MB file.

same model (Galaxy Tab 2) in this study, one as the measurement target and the other as a competitor. Two tablets were placed 3 meters away from each other and associated with different APs with RSSI around -40 dBm. We measured the transmission time for downloading a 33 MB file over 802.11n from a local file server. The competing traffic from the other tablet was a bulk download (94 MB file) via its associated AP from another local file server. We considered three cases in this study: a single device downloading alone over channel 1, two devices downloading over the same channel 1, and two devices downloading over different channels (measured device over channel 11 and competitor over channel 1). We repeated each case 5 times for both WiFi APs [6] and plotted the results in Figure 5.16.

From the results shown in the figure, the competing effect is visible when nearby devices are using the same channel even through different WiFi APs. By using different channels, such impact can be alleviated. This indicates that we need to consider the channel setting of nearby APs when selecting a target WiFi AP for offloading.

Third, we investigate the scenario where rate limiting policies are applied in the access network to control the transmission rate of each device, which is common in campus/enterprise environments. Figure 5.17 depicts the testing layout for this study to understand the effects of rate limiting on competing devices using the same WiFi AP. We set up a Linux laptop (DELL E4200) equipped with a NETGEAR WNA1100 wireless adapter to act as the 802.11n WiFi AP. To enable rate limitation for each device based on its IP address, we used the "tc" tool on Linux to manipulate traffic control setting. We cover the cases where different rate limit policies are applied to two competing devices, including 8 Mbps, 20 Mbps, and 24 Mbps, in addition to the case without rate limiting. We selected two tablets (Galaxy Tab 2) for this study, one as the measured target and the other as a competitor. We measured the transmission time for downloading a 33 MB file over 802.11n from a local file server. The competing traffic from the other tablet was a bulk download (94 MB file) via the same AP. We repeated each case 5 times and report the mean values with standard deviation for the measured device in Figure 5.18.

Our test data shows that the rate limit policies applied in the access network can affect the performance of an individual device in the competing traffic scenario. The effect is dependent on the exact rate limit in proportion to the maximum rate of a WiFi AP. As measured in the 'No Limit' case shown in Figure 5.18, the maximum transmission throughput for a single

[6] We used ASUS RT-AC68U and D-Link DWA-125 as the WiFi AP in this study.

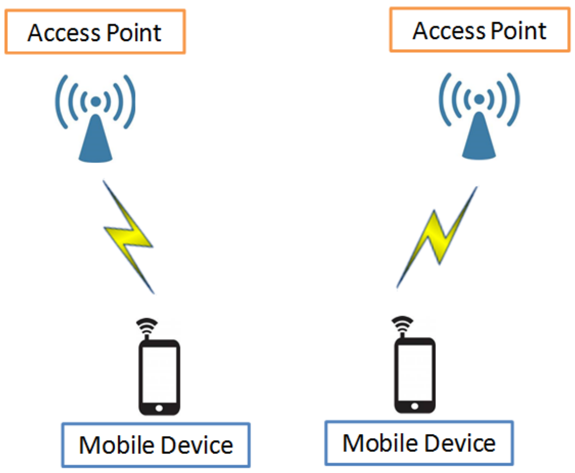


Figure 5.15: Test layout for competing case over different APs.

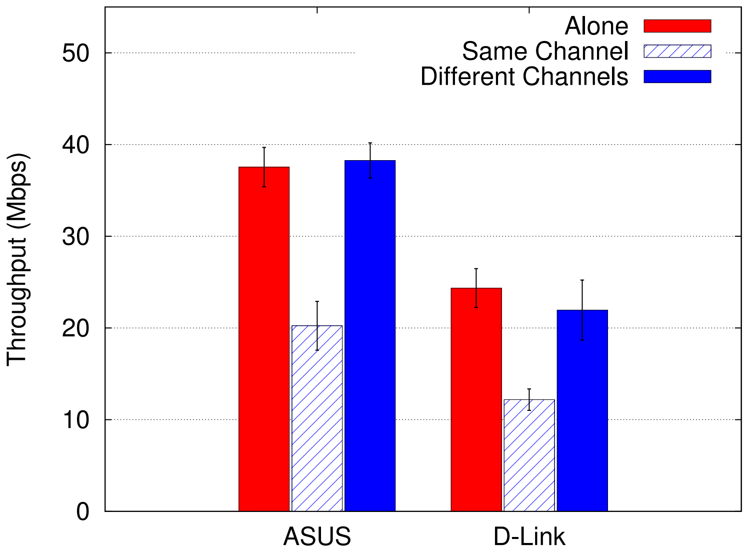


Figure 5.16: Throughput on the measured tablet in different cases.

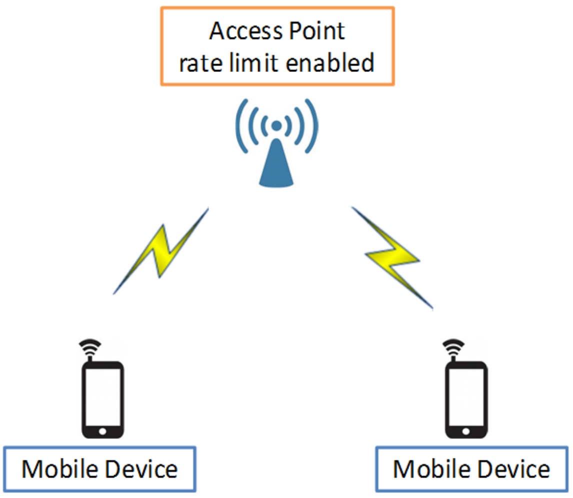


Figure 5.17: Test layout for competing case with rate limiting enabled.

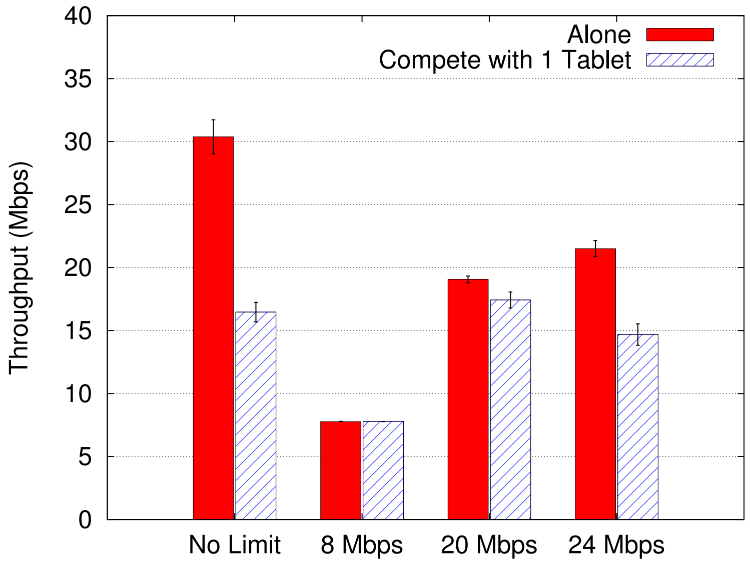


Figure 5.18: Competing scenario under different rate limit policies.

Galaxy tablet was ∼30 Mbps in our testing environment. If there was no rate limit, the throughput dropped to 16 Mbps when competing with the other tablet. When we limited the rate to 8 Mbps, the impact on throughput from the other tablet was negligible, and measured device did not experience visible throughput drop. By setting the rate limit to 20 Mbps per device, the interfere effect started to appear and the throughput dropped by approximately 10%. When we limited the rate further to 24 Mbps, the throughput of the measured device dropped by approximately 30%.

The observation indicates that network policies such as rate limitation are important context and we should consider them in making the WiFi offloading decision.

***Revisit WiFi Access Availability***

The measurement data discussed in Section 4.2.1 has shown that the openly accessible WiFi APs in metropolitan areas were very scarce, which can affect the design and deployment of WiFi offloading. We are interested in the current status and hence conducted a field study in Helsinki to acquire an updated view in 2014.

We selected Aleksanterinkatu, one of the most popular commercial streets in downtown Helsinki, for this measurement study. Figure 5.19 shows the map overview and our walking trails. We developed an Android application (Android 4.0+) to conduct WiFi scanning and evaluating the accessibility of WiF APs. The program will scan neighboring WiFi APs and perform 'Ping' test to a public reference server for all open APs that were detected in the scanning. We performed this measurement study in April 2014 using a smartphone (Galaxy S4) and a tablet (Galaxy Tab 2). During the test, each device was held by a person moving along the street at walking speed. Our tests cover both sides of the street and we summarize our results in Table 5.4.

Table 5.4: Availability WiFi access in downtown Helsinki.

| Detected APs | Open APs without Encryption | Accessible APs |
|---|---|---|
| 877 | 220 (25%) | 13 (**1.5%**) |

Based on the collected data, we detected 877 WiFi APs in total. Although 25% of the detected APs are open without encryption, the percentage of accessible APs was around 1.5%, which was very low [7]. We also

[7] The ground truth can be higher than this number as our field testing may not fully

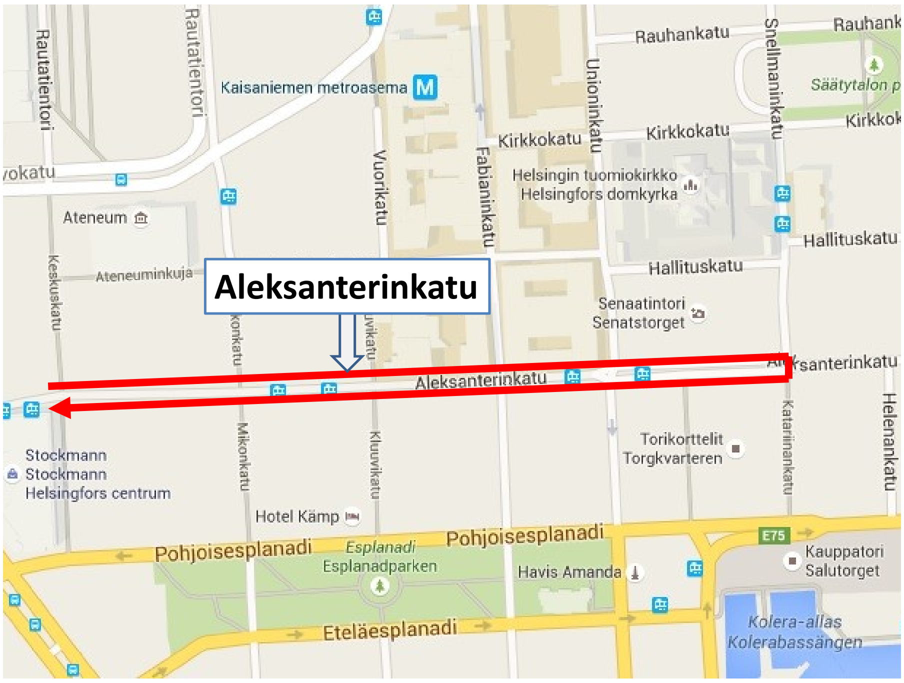


Figure 5.19: Walking trails in downtown Helsinki.

noticed that among the 'open' APs, many of them required only web-based activations without any password or user account. For example, we detected 27 WiFi APs using ESSID 'Helsingin kaupungin WLAN', which can provide free Internet access if we activated it through its web interface. Since user interactions (i.e., clicking on Web browser) are needed for this type of WiFi access networks, we do not include them in to the 'Accessbile APs'.

Our measurement data shows that the WiFi access availability is still quite limited. Due to such low ratio, it will be challenging if we perform WiFi offloading only using open APs. Besides collaborative design as discussed in Chapter 4, we need to consider how to improve deployability of a proposal to allow operators and WiFi providers to adopt the scheme.

### 5.2.2 Measurement insight

Our measurement study is motivated by the following questions: 1) Is the performance of latest WiFi technology good enough to support mobile traffic offloading, especially compared with the 3GPP 4G/LTE access

cover the whole spectrum in this area.

technologies? 2) What types of contextual information are important in WiFi-based traffic offloading? 3) What is the impact of such contextual information? Can we efficiently utilize these information for assist traffic offloading?

Regarding these research questions we investigated through live experiments, the key observations include:

- Given the advance of WiFi technologies such as 802.11ac, WiFi-based traffic offloading is still a viable and economical solution to alleviate the load pressure on the incumbent cellular networks. At the same time, the transmission rate in cellular access has been improved significantly. Unlike the performance reported in the early study [5], commercial LTE networks can offer good throughput for downlink transmission and are also energy efficient. Although WiFi can offer higher link access bandwidth, we should be prudent to use WiFi networks to offload mobile data traffic.

- In WiFi offloading, contextual information from end users is as important as that from the network side. Among various context, the WiFi signal strength (reflected in RSSI) from the user side is crucial in WiFi offloading. Regarding the network side, the number of active users associated with AP, channel settings, and rate limit policies are also important. All these context can affect performance and energy consumption in WiFi offloading.

- An end user may suffer from serious performance degrade and high energy consumption by choosing a ‘sub-optimal’ WiFi AP (e.g., with weak signal strength or overloaded by competing traffic from other devices) for offloading. This directly affects the user experience and hinders the adoption of WiFi offloading solution. We need to collaboratively utilize the available context from both end users and the network side to guide the traffic offloading process.

## 5.3 System Design

Offloading mobile traffic to WiFi networks (i.e., WiFi Offloading) is a cost-effective technique to alleviate the pressure on mobile networks for meeting the surge of data capacity demand. However, existing proposals from standards developing organizations (SDOs) and the research community are facing a deployment dilemma, either due to overlooking device limitations, lack user incentives, or missing operator support. Given the complexity of network deployment and device limitations, solutions that neglect these factors can impede their deployment.

In addition, despite the lengthy process required for standardization [17], proposals from SDOs and research communities typically address different segments of WiFi offloading such as offloading efficiency [5, 6, 7, 8], energy efficiency [16, 11], user incentives [9], and operator supports [12, 26]. Those solutions are often isolated from each other, built from scratch for incumbent legacy hardware, or overlooking practical limitations of mobile devices and operational environments. Such issues make the solution deployment a formidable task in today's fast evolving mobile networks.

Given that WiFi networks can be considered as an integral part of the upcoming 5G mobile architecture [31], this provides an opportunity for WiFi offloading in that cellular and WiFi accesses can be unified and managed through the same authentication and billing system.

We propose SoftOffload, an open-source platform for WiFi offloading to tackle the deployment challenge. Our solution leverages the programmable feature of software-defined networking (SDN) to enhance extensibility and deployability in a collaborative manner.

### 5.3.1 Design overview and principles

Benefiting from the features of SDN in terms of programmability and openness, SoftOffload aims to improve extensibility and deployability. Figure 5.20 presents an overview of our proposal in an SDN-enabled mobile wireless access. The key components of SoftOffload include an SDN-based central controller, local agents, and an extension module for mobile devices. We use SDN-enabled switches (e.g., with OpenFlow support) for monitoring and managing traffic flows.

We highlight the design principles of SoftOffload as follows:

- Collaborative and efficient management - Based on a hierarchical design, our solution efficiently controls cellular and WiFi networks in a collaborative manner. By splitting offloading functionality between central controller and offloading agents deployed at the network edge, SoftOffload strikes a balance between responsiveness and global control.

- Context awareness - To benefit both network operators and end users, our platform extracts and synthesizes essential context information from wireless networks and mobile devices. SoftOffload utilizes these context and dedicated offloading algorithms to make balanced off-

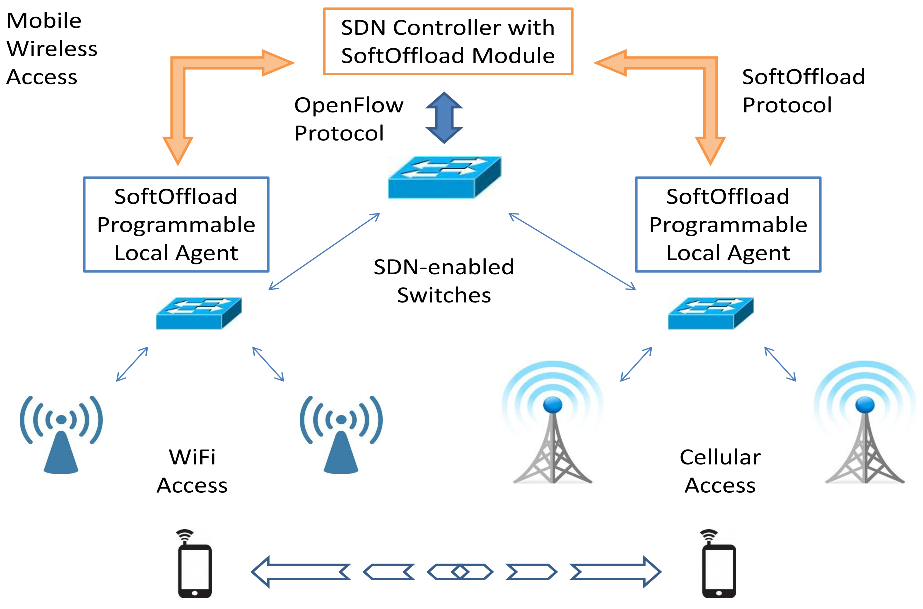


Figure 5.20: SoftOffload overview [30].

loading decisions that improve both offloading efficiency and energy saving.

- Openness and extensibility - SoftOffload builds on open standard protocols (e.g., OpenFlow) and benefits from the programmable feature of SDN. We aim at a light-weight overlay design for wireless networks without adding extra modifications or requirements towards existing standards. Through the integration of our context-aware offloading algorithm, as a good example, we demonstrate how to add innovative applications on top of this open and extensible platform.

The value of SoftOffload comes from its open and extensible design which facilitates development of new extensions/applications for different wireless access environments. The open-source oriented design combined with standardized SDN protocols encourage community contributions.

### 5.3.2 Software-defined collaborative platform

We design SoftOffload as an extensible platform. The goal is to harness the programmability and openness of SDN and thereby enhance the deployability in multi-access wireless environments.

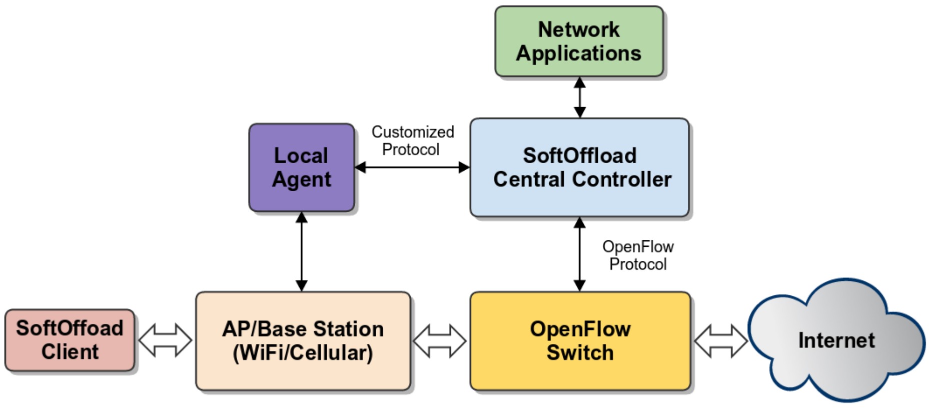


Figure 5.21: System architecture.

### *Architecture and Key Components*

Figure 5.21 illustrates the overall architecture of SoftOffload, which consists of three major components: a central controller, local agents, and a client extension for mobile devices. SoftOffload adopts a hierarchical design to reduce the signaling and latency overhead between a centralized controller and local agents. The essential offloading functions such as traffic monitoring, context processing, and decision making are built on top of the centralized controller as network-level SDN applications. It uses OpenFlow to manage OpenFlow-enabled switches and acquire flow information in the managed network. Local agents are deployed as an integrated module on wireless access points and cellular base stations, responsible for collecting flow statistics, performing localized flow processing, and communicating with both central controller and client extension.

The SoftOffload client extension can be deployed on mobile devices as an application or an integrated service offered by the mobile operating system. It enables SoftOffload to collect user-side context such as WiFi signal strength and mobility status of users. By exploiting the rich context on mobile devices, SoftOffload can attain better visibility towards the condition of wireless networks, which are important for making balanced offloading decisions. With such enhanced functionality offered by the client extension, SoftOffload can collaboratively solve problems that are difficult by using only network knowledge and observations.

As SDN paradigm is gradually accepted by mobile industry, the SDN-based design adopted by SoftOffload not only encourages its acceptance by

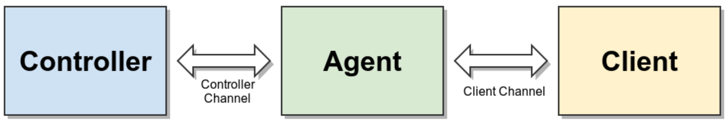


Figure 5.22: Communication channels in SoftOffload.

the mobile carriers, but also opens up deployment opportunities by new players such as cloud service providers that are interested in becoming mobile virtual network operators (MVNO) in partnership with WiFi service operators. In addition, SoftOffload can be easily extended by adding new applications on the top of the SoftOffload controller or through extensions for the SoftOffload client and local agents. With this extensible design, optimization schemes and offloading policies can be conveniently added to support for different scenarios.

### *Communication and Operational Schematics*

SoftOffload utilizes two communication channels to coordinate controller, local agents and clients. Figure 5.22 illustrates the 'Control Channel' and 'Client Channel' used in SoftOffload. The 'Control Channel' is used by central controller and local agents to exchange control messages and status reports. For example, when an agent detects a new mobile device associated to the network, it sends a status update message to central controller to report this event.

The 'Client Channel' is to enable data collection and client management. Rather than creating a separate channel between the central controller and each client, we use local agents as a proxy for communication. Messages from the SoftOffload client are passed first through 'Client Channel' to a corresponding local agent. The local agent will process the messages and forward, if needed, post-processed messages to the central controller via 'Control Channel'. Similarly, the central controller also uses local agents to send requests to the managed clients. In this way, the central controller is not directly exposed to clients. By using local agents to pre-process messages and block suspicious flows, we avoid overloading the central controller and backhaul link, and thereby enhance robustness and security of the system.

Figure 5.23 illustrates the operation sequence to conduct WiFi offloading in a typical scenario across two access points. To initialize traffic offloading, a central controller monitors its managed network by collecting network statistics from OpenFlow switches. If it detects unbalanced or congested traffic patterns in a specific access network (WiFi or cellular), the controller triggers the offloading and follows the steps shown in the figure.

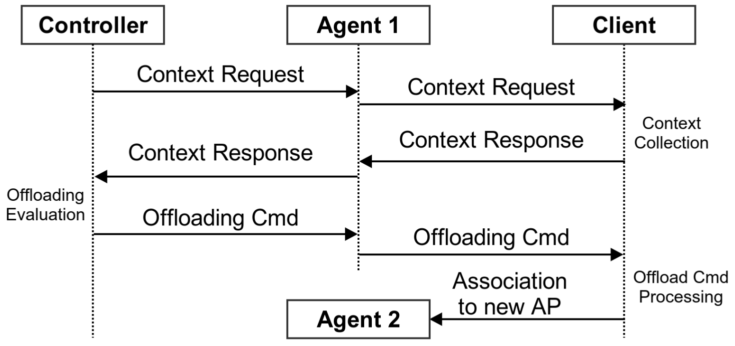


Figure 5.23: Operation sequence.

In brief, the central controller manages the entire network. It is responsible for tracking all the clients and agents in the managed network, collecting traffic information from network devices, and further making offloading decisions based on inputs from various sources. In addition to obtain flow information from switches, SoftOffload local agents monitor the association status of mobile devices and serve as an information channel between the central controller and mobile devices. The client extension collects context information from the device and access environment, including user movement and signal strength of neighboring WiFi APs. Central controller collects such information to perform offloading management.

### *Context-aware and Collaborative Offloading*

A key challenge in WiFi offloading is that valuable contextual information is often distributed across network entities and end users. Such uneven distribution implies that the contextual information possessed by each side along is insufficient and could mislead the offloading process.

On the network side, the traffic load and conditions of managed entities (e.g., router, APs) can be acquired to assist traffic offloading. For example, operators can decide to trigger traffic offloading when a cellular base station is overloaded by several active users. However, it is difficult, if not impossible, for operators to capture the instant local context perceived by those mobile users (e.g., signal strength of nearby WiFi APs).

On the other hand, mobile device can acquire local context by using

**Algorithm 2** SoftOffload context-aware offloading algorithm.

```
 1: INPUT: The set of access points and access switches $W$
 2: INPUT: The total bandwidth $maxBandwidth_i$ for an entity $i$
 3: INPUT: The current bandwidth $currentBandwidth_i$ for an entity $i$
 4: INPUT: The set of clients $C$
 5: INPUT: Constant coefficient $k$ and detection interval $v$
 6: while TRUE do
 7:    for each $w \in W$ do
 8:       Detect Traffic Load on $w$
 9:       if $currentLoad_w \geq k \cdot maxBandwidth_w$ then
10:          $C \leftarrow$ Choose Offloading Client Set on $w$
11:          for each $c \in C$ do
12:             Evaluate Offloading Target (Formula 5.1) for $c$
13:          end for
14:       end if
15:    end for
16:    Wait for Detection Interval $v$
17: end while
```

WiFi scanning and embedded sensors on their devices, but they are hardly aware of the condition on the network side (e.g., congestions on the aggregated router) before connecting to the access network. A typical example is that, even if a WiFi AP is relatively close to the user, showing good signal strength, this AP can be heavily loaded by other devices. If the user chooses to offload traffic using this congested AP (with good signal strength though), the outcome can be suboptimal.

By observing the importance of collaborative knowledge from both wireless networks and end users, SoftOffload exploits the context awareness in making offload decisions. It considers several factors based on the collaboratively collected context, such as network bandwidth and WiFi signal strength.

We describe the decision making process of SoftOffload in Algorithm 2. The process runs continuously (e.g., as a daemon process) and controls all the managed entities (i.e., access points and base stations) denoted $W$, and clients (i.e., mobile devices) denoted as $C$.

The key steps in the decision making process include 'Detect Traffic Load', 'Choose Offloading Client Set', and 'Evaluate Offloading Target'. We illustrate each step as follows:

- Traffic Load Detection – In SoftOffload, load detection is performed by a monitoring process that follows a set of pre-defined rules (e.g., load threshold and number of active users) to trigger the offloading procedure. SoftOffload uses $k \cdot maxBandwidth_w$ as the traffic threshold where $k$ is a constant coefficient to allow adjustment of the threshold. For example, if the traffic load on an entity (e.g., a WiFi AP) surpasses the pre-defined threshold and the congested state lasts longer than a pre-defined period, SoftOffload can trigger offloading according to its load detection rules.

- Choose Offloading Client Set – Once the offloading procedure is triggered, SoftOffload starts choosing clients (i.e., mobile devices) associated to the entity (e.g., WiFi AP) as offloading candidates. Based on the observation of rate limiting effects (Section 5.2.1), SoftOffload adjusts its selection mechanism depending on the rate limiting policies. If there is no rate limit, SoftOffload will choose the clients with the highest bandwidth utilization to offload. If the rate limiting policies are applied, each client only occupies a fraction of the total bandwidth in the access network. To avoid the problem of offloading a large amount of clients with ongoing traffic load, which is inefficient and creates extra overhead, SoftOffload will choose those inactive clients associated to the access network as candidates, rather than forcing active clients to migrate away.

- Evaluate Offloading Target – By observing the importance of a balanced decision that can benefit both network operators and end users [16], SoftOffload collaboratively collects contextual information from local agents and client. To select the best possible offloading target (i.e., WiFi AP), SoftOffload utilizes a novel context-aware decision formula, which guides SoftOffload to make optimal decision.

  The proposed context-aware formula normalizes different factors and produces an offloading metric value. SoftOffload runs the formula for all available APs (including the current one to which the client connects), ranks the metric values, and selects the most suitable AP as the offloading target.

  The core of the formula is shown below:

$$(1-P) \cdot \frac{estimatedBw}{\arg\max estimatedBw} \cdot \frac{restBw}{totalBw} \cdot \mu - c \cdot O \tag{5.1}$$

  The throughput penalty $P = e^{-k_0(S-k_1)}$ is an exponential coefficient to reflect the impact of signal strength. *estimatedBw* is the estimated

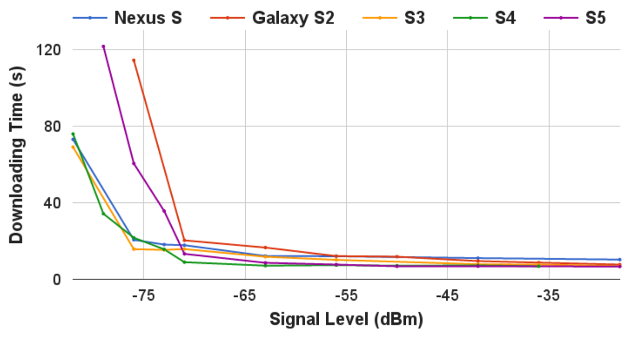


Figure 5.24: Downloading time [28].

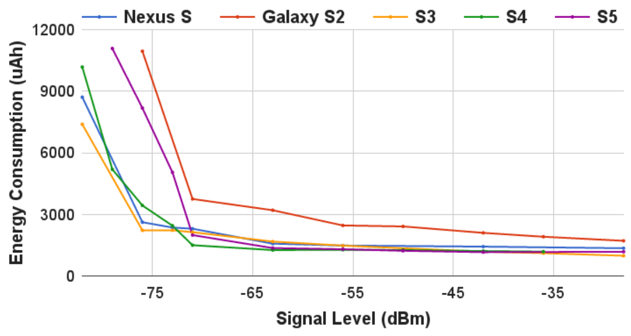


Figure 5.25: Energy consumption [28].

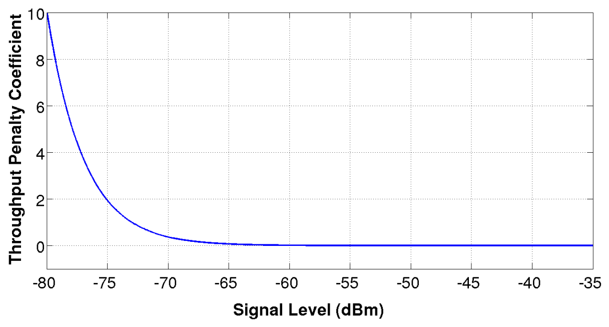


Figure 5.26: Signal level and penalty coefficient $P = e^{-\frac{1}{3}(S-(-73))}$ [28].

bandwidth a client can obtain from a target AP, and arg max $estimatedBw$ is the maximum value of the estimated bandwidth among all available APs that SoftOffload considers for offloading. $restBw$ and $totalBw$ show the remaining and total bandwidth, respectively, on a target AP. We use $\mu$ to evaluate mobility effects on potential throughput, and $O$ as an indicator function for offloading overhead. SoftOffload assigns $O = 1$ for all the available APs other than the AP to which the client currently connects, to reflect the cost involved in offloading (e.g., switching networks, extra scanning). The value of $O$ becomes 0 when SoftOffload evaluates the same AP that the client currently connects to. $c$ is a weight factor that provides flexibility to adjust offloading metric by taking into account network policy, user preference and device limitation.

According to our measurements (Section 5.2.1) as highlighted in Figure 5.24 and 5.25, we observe a steep drop of performance together with high energy consumption when the signal level is decreasing down to -73 dBm. Because the energy consumption is proportional to the transmission performance, we exploit this feature to reflect the impact of signal strength and integrate context awareness into SoftOffload. We derive a penalty factor $P = e^{-k_0(S-k_1)}$ based on the observation and assign $k_0 = \frac{1}{3}$, and $k_1 = -73$. As shown in Figure 5.26, this penalty coefficient $e^{-k_0(S-k_1)}$ on signal level $S$ matches coherently to the performance and energy consumption in Figure 5.24 and Figure 5.25, respectively.

Once the decisions are made, SoftOffload will notify the chosen clients to offload their data traffic to the target WiFi access network.

## 5.4 Implementation and Experiments

In this section, we present the SoftOffload prototype and illustrate how SoftOffload enables context-aware WiFi offloading in practice. To demonstrate the efficacy of our solution, we further evaluate SoftOffload through experiments in our testbed.

### 5.4.1 System prototype

We have developed a prototype system based on the SoftOffload design. Figure 5.27 shows the key components and schematics of the implemented

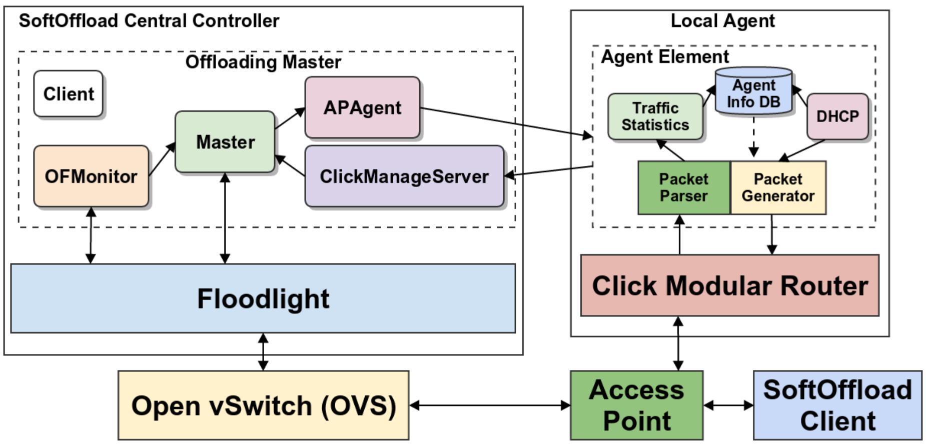


Figure 5.27: System implementation.

prototype. The SoftOffload central controller is built based on the Floodlight OpenFlow controller [8], which manages the entire network and makes offloading decisions. The local agent is developed based on Click modular router [281] and deployed on each AP. It monitors the corresponding AP interface and reports association events to the central controller. The client extension is implemented as an Android application, which listens on a specific UDP port for controlling messages from the controller and local agents, and reacts to the requests. In addition, we use Open vSwitch (OVS) [9], which provides OpenFlow support for our platform.

### *Central Controller*

We implement SoftOffload central controller with Floodlight OpenFlow controller in Java. The core of the central controller is 'Offloading Master', which is an independent network application running on top of Floodlight.

'Offloading Master' manages the offloading process and utilizes OpenFlow APIs to control and monitor the OpenFlow-enabled devices. As shown in Figure 5.27, 'Offloading Master' consists of five modules: *Master*, *OFMonitor*, *APAgent*, *Client*, and *ClickManageServer*.

- *Master* – The 'Master' module is the logical "brain" of the central controller. It tracks all the clients and agents in the network, and

[8] `http://www.projectfloodlight.org/floodlight/`
[9] `http://openvswitch.org/`

makes offloading decisions based on the network policies and contextual information collected from various sources.

- *OFMonitor* – The ‘OFMonitor’ runs as a separated thread and monitors corresponding OpenFlow switches. If it detects traffic loads or specific traffic patterns on its monitored switches, it will inform the ‘Master’. The ‘Master’ and the ‘OFMonitor’ use the Floodlight APIs and OpenFlow protocol to communicate with underlying switches.

- *APAgent* – ‘APAgent’ is an abstract module that represents a local agent and stores agent information. It contains necessary information about a WiFi access point like SSID, BSSID and traffic loads, and can be extended for different network types such as cellular access. The ‘APAgent’ also records all the connected mobile clients and provides necessary functions for the ‘Master’ to interact with access points (e.g., send messages to agents, add or remove clients). The central controller initializes an ‘APAgent’ instance for each local agent when the system starts up.

- *Client* – Similar to ‘APAgent’, SoftOffload uses ‘Client’ to represent mobile devices. When a mobile device connects to the network, the ‘Master’ will initialize a Client instance and record its information (e.g., downloading rate, connection time, and to which APAgent it connects).

- *ClickManageServer* – ‘ClickManageServer’ listens on a particular port for exchanging messages with the SoftOffload local agents. It collects messages from agents and invokes corresponding functions in the ‘Master’. For example, if it receives messages reporting an AP’s downstream rates, it will inform the ‘Master’ to update the records for this AP.

#### *Local Agent*

We implement the SoftOffload local agents with *Click* modular router, which is a programmable software architecture for building flexible and modular routers to forward packets. *Click* provides various modules, referred to as elements, to support different management functions such as packet classification, routing, and monitoring. The elements can be assembled together to accomplish demanding tasks. *Click* uses a configuration file with particular syntax to mandate how packets are processed by a Click-

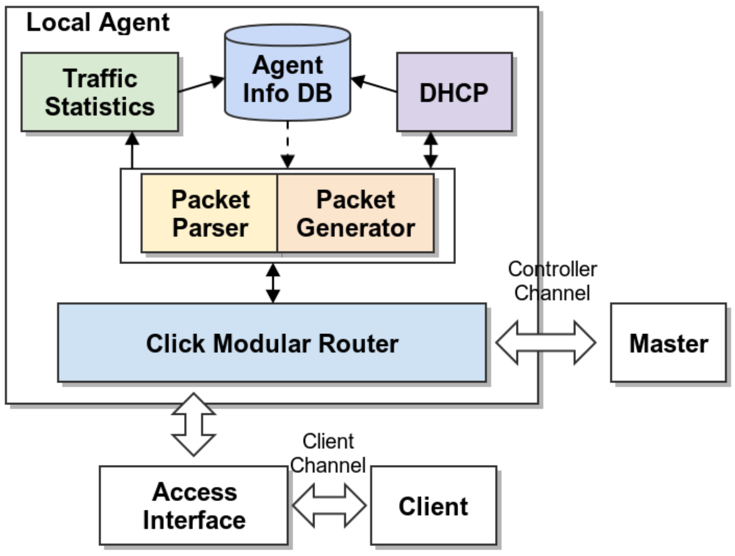


Figure 5.28: Implementation of local agents.

enabled router. The configuration specifies which elements are used in the router and in what sequence they operate on the corresponding packets.

For the features *Click* provides, we use it to build our local agents. Figure 5.28 highlights the structure of the Click-based local agent. It typically runs in monitoring mode on an access interface, which allows it to use Click to parse all the packets appearing on this interface and generate new packets.

Following our design, a 'Local Agent' handles client association and informs the central controller via the controller channel about client connection/disconnection. To enable fine-grained control on clients, we develop a customized *DHCP* module. Besides allocating IP addresses to clients, when a client obtains an IP address, it informs the 'Offloading Master' for this new connection.

An agent also performs 'keep-alive' testing for each client associated with the AP by periodically sending Ping packets (ICMP echo requests) to each client. If a client does not respond for five consecutive Ping packets [10], the agent will consider it as disassociated, and notify the central controller about this incident. SoftOffload adopts this mechanism for detecting client hand-off events, which helps the central controller to maintain a real-time and accurate view of the managed network.

[10] The 'keep-alive' threshold can be adjusted through configuration.

In order to record clients and their traffic information, we implement an *Agent Info DB* module and use a particular data structure to represent mobile users. When a client is connected to an access point, the corresponding local agent will first initialize a *Client* instance for it, and record necessary information about this client (e.g. IP and MAC address). By tracking all the clients, 'Local Agents' can provide efficient and flexible client management.

***SoftOffload Client***

We implement an Android application as prototype for SoftOffload Client, which is an important piece of SoftOffload to enable collaborative offloading. The prototype application currently supports two major functions:

- WiFi scanning and reporting – Upon receiving a scanning request, our application will conduct WiFi scanning and report the information of neighboring APs (i.e., MAC addresses, ESSIDs, signal levels) to the controller. We implement this function using the ***android.net.wifi*** package[11].
- Switching access networks – As the core function for mobile devices to participate in WiFi offloading, the client application enables a mobile Android device to disconnect from the existing AP and associate with the target AP by following the commands of the SoftOffload central controller. We implement this function also with the ***android.net.wifi*** package.

In addition, our client application is able to report the list of active applications to SoftOffload controller as an extra contextual input. SoftOffload may use this information to assist its decision making. For example, if a controller detects that YouTube is actively running on a client, it may consider this client as a potential candidate for offloading, since this client is of higher probability to generate video streaming traffic. We note that it is difficult to predict traffic load based merely on the application list. This functionality covers only a limited set of applications and SoftOffload currently treats such context only for reference purpose.

***Communication Protocol***

[11] Android API: http://developer.android.com/reference/packages.html

SoftOffload utilizes the OpenFlow protocol through Floodlight to communicate with OpenFlow-enabled switches. In order to enable efficient communication between central controller and other SoftOffload components (i.e., local agents and clients), we define and develop a customized SoftOffload protocol. The protocol runs on top of UDP to support on-demand and flexible communication between the SoftOffload components. Figure 5.29 illustrates the message format used by SoftOffload, which consists of two major fields: an optional 'Forward Field' and a mandatory 'Message Field'.

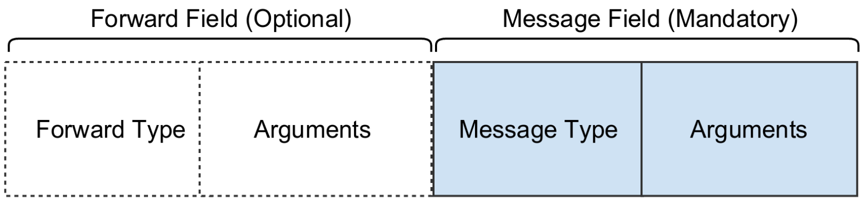


Figure 5.29: SoftOffload message format.

As SoftOffload adopts a hierarchical structure in which local agents serve as proxies between central controller and clients, it enables message forwarding by using the 'Forward Field' to convey necessary information to local agents. Therefore, all messages exchanged between the central controller and clients must use this field. Only the messages directly sent by local agents do not use this optional field. Table 5.5 presents the message detail for the 'Forward Field'. The 'Message Field' is used by the prototype system to enable collaborative traffic offloading. We list messages details of 'Messsage Field' in Table 5.6 and Table 5.7, for Agent-Controller interaction and Client-Controller interaction, respectively.

We illustrate in Figure 5.30 an example of message exchange using both 'Forward Field' and 'Message Field' to request WiFi scanning (SCAN_AP). As shown in the figure, the controller first sends a SCAN_AP request to the client through a local agent. The agent check the 'Forward Field' to fetch the MAC address of the client, and then deliver the rest of the message (i.e., 'Message Field') to the target client. After WiFi scanning, the client responds with its scanning results and points to its controller, the corresponding agent will parse the message and forward the AP_STATS to the controller.

Table 5.5: Message specification of Forward Field.

| **Type** | **Arguments** | **Direction** | **Description** |
|---|---|---|---|
| TO_CLI | $mac\_addr_{client}$ | Controller to Client | For controller to send messages to a client |
| TO_CTR | None | Client to Controller | For a client to send messages to its controller |
| TO_AGENT | None | Controller/ Client to Local Agent | For controller or a client to send messages to an agent |

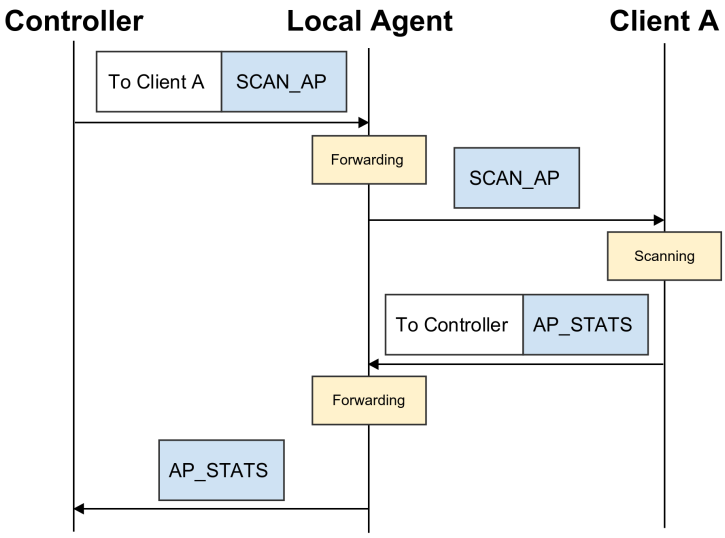


Figure 5.30: A message exchange between Controller and Client.

Table 5.6: Agent-Controller messages of Message Field.

| **Type** | **Arguments** | **Direction** | **Description** |
|---|---|---|---|
| REMOVE_CLI | $mac\_addr_{client}$ | Controller to Agent | Inform agent to remove a client record |
| CHANGE_CH | $new\_channel$ | Controller to Agent | Inform agent to change to a different channel |
| ADD_CLI | $mac\_addr_{client}$ $ip\_addr_{client}$ | Agent to Controller | Inform controller to add a new client connecting to this agent |
| AGENT_RATE | $up\_rate_{agent}$ $down\_rate_{agent}$ | Agent to Controller | Inform controller the traffic rate of this agent in recent time slot |
| CLI_RATE | $mac\_addr_{client}$ $up\_rate_{client}$ $down\_rate_{client}$ | Agent to Controller | Inform controller the rate for a client in recent time slot |
| DISCON_CLI | $mac\_addr_{client}$ | Agent to Controller | Inform controller that a client is disconnected |

Table 5.7: Client-Controller messages of Message Field.

| **Type** | **Arguments** | **Direction** | **Description** |
|---|---|---|---|
| SWITCH_AP | *ssid*, *bssid*, *auth_method*, *auth_password* | Controller to Client | For controller to send messages to a client |
| SCAN_AP | None | Controller to Client | Inform client to scan nearby APs |
| QUERY_APP | None | Controller to Client | Request client to report list of running apps |
| AP_STATS | *ssid*, *bssid*, *RSSI* | Client to Controller | Report to controller the scanning results |
| APP_STATS | *app_name* | Client to Controller | Report to controller the list of running apps |

***Enabling Context-aware Collaborative Offloading***

SoftOffload utilizes a set of functions to enable context-aware collaborative offloading. Figure 5.31 presents an overview of the offloading process. We illustrate here how these functions are implemented in the prototype, including traffic monitoring, context collection, and movement estimation:

- Traffic Monitoring – SoftOffload utilizes the *OFMonitor* module we implemented to monitor access switches with the OpenFlow protocol. The *OFMonitor* collects traffic statistics at fixed interval and estimates the transmission rate. To avoid jitter effects in the traffic sampling, we implement a pending counter mechanism for monitoring. With this scheme, we use a counter to track how many times the average rate goes over the threshold. If the average rate surpasses the threshold and then drops down, the counter will not be reset to zero immediately. If the high rate appears again within a pre-defined pending time window, the counter will be resumed and start from its previous pending value, instead of from zero. In case the average rate drops down for longer than the pending time window, the counter will be reset to zero.

- Context Collection – Once SoftOffload triggers offloading and has chosen the candidate clients, it starts to collect contextual information

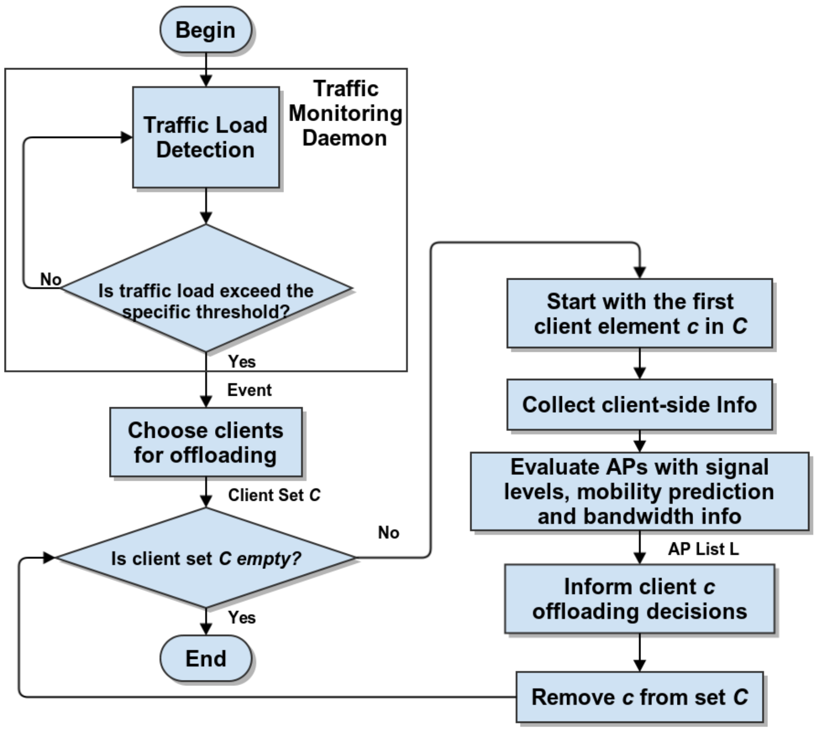

Begin
Traffic Monitoring Daemon
Traffic Load Detection
No
Is traffic load exceed the specific threshold?
Yes
Event
Choose clients for offloading
Client Set *C*
Is client set *C empty?*
No
Yes
End
Start with the first client element *c* in *C*
Collect client-side Info
Evaluate APs with signal levels, mobility prediction and bandwidth info
AP List L
Inform client *c* offloading decisions
Remove *c* from set *C*

Figure 5.31: Offloading procedure [28].

from both network and end users. The central controller will spawn a process requesting each client to report its WiFi scanning results and collecting bandwidth information from the SoftOffload-enabled APs that appear in the scanning results. Each local agent will report traffic statistics to the central controller periodically. SoftOffload stores those statistics messages and use the collected information to estimate bandwidth utilization for each AP. We note that the process of collecting client information as depicted in Figure 5.31 is serial for the current prototype implementation. A real deployable system must do it in parallel in order to make the system scale.

Regarding the bandwidth estimation as described in the context-aware formula:

$$(1-P)\cdot\frac{estimatedBw}{\arg\max estimatedBw}\cdot\frac{restBw}{totalBw}\cdot\mu - c\cdot O$$

If there is no rate limit policy, $restBw$ is equal to $estimatedBw$. However, in the case of rate limit policies being applied, $restBw$ is often different from $estimatedBw$. The $\frac{estimatedBw}{\arg\max estimatedBw}$ ranks each candidate AP against all available APs. The $\frac{restBw}{totalBw}$ indicates the bandwidth utilization on this specific AP. The value of $\frac{estimatedBw}{\arg\max estimatedBw} \cdot \frac{restBw}{totalBw}$ reflects the normalized bandwidth capacity and utilization for an AP. Regarding overhead, an empirical value 0.2 is used for $c$ in SoftOffload prototype.

- Movement Estimation – Besides signal strength, user's mobility status is another important context. To avoid the overhead introduced by complex prediction schemes, SoftOffload only uses short-term estimation to support its decision making. In the current prototype, we develop a simple and low-overhead method for SoftOffload, in which a client conducts WiFi scanning for three times and central controller will make a short-term movement estimation based on the reported scanning results.

  SoftOffload uses a weight value $\mu$ $(0 < \mu \leq 1)$ to convey the influence of the user movement. The implemented movement estimation scheme calculates the $\mu$ for each AP by comparing the results collected from three consecutive scanning, namely $s_1$, $s_2$, and $s_3$. Depending on the pattern revealed by the scanning results, we derive a corresponding value for $\mu$. We illustrate the pattern classification as follows:

  1. Moving away:
     - $(s_3 \leq s_2 \leq s_1) \wedge (s_3 < s_1)$: This pattern reveals that the signal strength for a WiFi AP is clearly decreasing. SoftOffload regards this as an indication that the client is moving away from the AP, and thereby assigns a small value 0.7 to $\mu$ in the existing implementation.
     - $(s_2 > s_1) \wedge (s_3 < s_1)$: This pattern indicates a fluctuation, where the signal strength first increases as shown by $s_2 > s_1$, but finally drops (i.e., $s_3 < s_1$). SoftOffload considers this case as moving away but assigns value 0.8 to $\mu$ in order to imply the fluctuation effect.
  2. Approaching:
     - $(s_3 > s_2 > s_1) \wedge (s_3 > s_1)$: This pattern shows a clear trend that the client is approaching this specific AP. In this case, SoftOffload assigns the maximum value 1 to $\mu$.

  - $(s_2 < s_1) \wedge (s_3 > s_1)$: This patterns indicates a fluctuation, in which the signal strength first drops $(s_2 < s_1)$, but finally increases $(s_3 > s_1)$. SoftOffload assigns value 0.9 in this case to express the fluctuation effect.

3. Other conditions:
   If the pattern indicated by scanning results does not fit in the categories listed above, SoftOffload assigns a medium value 0.85 to to $\mu$. This category covers the case that no clear movement trend is detected based on scanning results. To avoid over-penalizing, the value for this category is between the maximum (for approaching case) and the minimum (for moving away case). For example, $s_1 = s_3$ can be classified into this category as the client is likely to be still around this AP.

In brief, SoftOffload collaboratively collects necessary context from both the managed network and end users. It utilizes the proposed context-aware offloading algorithm to calculate metric values and rank candidate APs. Based on the comprehensive knowledge from various sources, SoftOffload makes a balanced decision and notifies the selected clients to offload their traffic to the target WiFi APs.

### 5.4.2 Experimental evaluation

To demonstrate the efficacy of SoftOffload, we evaluate the prototype system through two sets of experiments. First, we verify the feasibility of SoftOffload by testing WiFi offloading in both static and mobile scenarios. Second, we measure the system overhead for controller and end devices to support WiFi offloading.

#### ***Testbed***

As indicated in a study on mobile data usage [278], home and office are still the primary locations for using mobile devices such as laptops, eReaders, tablets, and smartphones. The content-rich applications used in those indoor environments represent the dominant mobile data generated by users. Therefore, we focus on indoor environment for our experiments. Figure 5.32 shows the testbed we set up in the laboratory for evaluating SoftOffload.

We run SoftOffload central controller on a Linux desktop machine with Intel E8500, 6 GB RAM, and Ubuntu 14.04 of kernel version 3.13.32. We install Open vSwitch (v2.3.0) on another desktop machine with the same

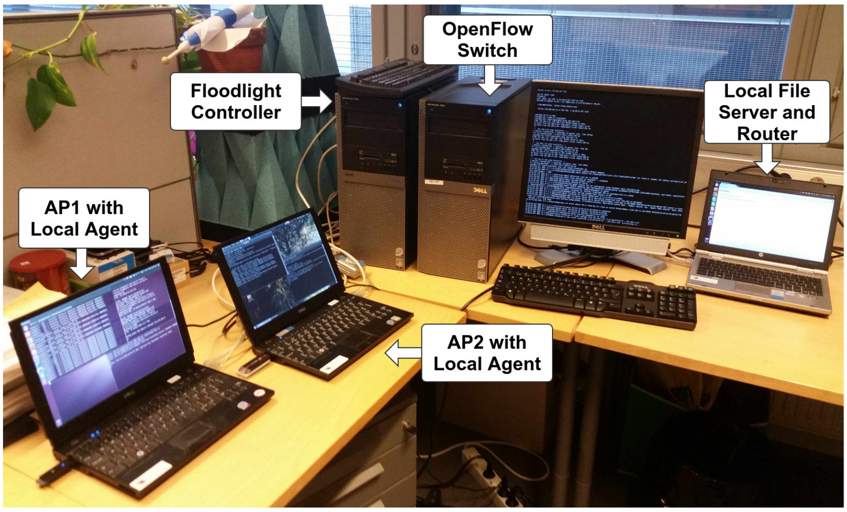


Figure 5.32: Experimental testbed [29].

hardware to support OpenFlow for our experiments. We use two Linux laptops with hostapd [12] and USB WiFi adapters to act as WiFi access points, and integrate SoftOffload local agent on them. We also use one laptop machine with an SSD drive to act as a local file server.

***Context-aware WiFi Offloading***

To validate context-aware offloading of SoftOffload, we first measure the offloading performance in a controlled environment where we impose rate limits on the WiFi access. These measurement results serve as the benchmark. We then repeat the same test cases with SoftOffload decision module and record the decisions made by SoftOffload to compare with the benchmark. To concentrate on the performance impact of context-aware offloading, we do not consider user movement in this comparison test and all tested devices are kept static nearby the WiFi APs (with RSSI of -40 dBm, AP operating in channel 6).

The network topology of this experiment is shown in Figure 5.33. We set up two WiFi APs (AP1 and AP2) and enable rate limits with tool “tc” on Linux. We limit the downlink rate of AP1 to 8 Mbps. We adjust the downlink rate on AP2 to cover 4 Mbps, 8 Mbps, 16 Mbps, and 24 Mbps. For the test client device, we use tablets (Galaxy Tab 2) running SoftOffload client application. For the controlled tests, we develop a dedicated

[12] `https://w1.fi/hostapd/`

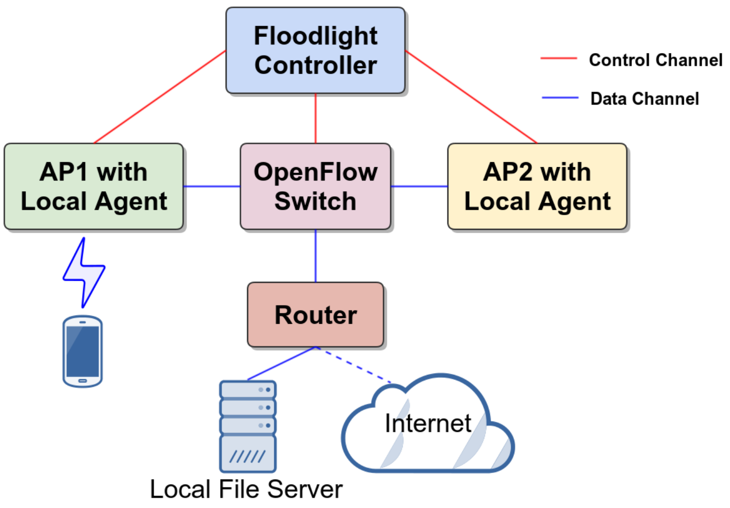


Figure 5.33: Network topology for offloading tests.

function on SoftOffload client to download files from the local file server. The downloading process will pause shortly when the client receives an offloading command from central controller, and then resume the downloading once the device connects to another AP.

To obtain comparison benchmark, the client is forced to switch to another AP regardless of the bandwidth context. Once the system detects that the traffic load goes above 70% of the available bandwidth on AP1, it will force the client to switch to AP2. We repeat the measurement 3 times for all the cases and report the results in Table 5.8. We use three files of size 33 MB, 65 MB, and 141 MB for testing. The 'Baseline' shows the downloading time by using AP1 (8 Mbps) without offloading.

As shown in the case of 4 Mbps, if an offloading decision is made without considering the bandwidth of the candidate AP (AP2 in our case), users can experience performance degradation. Even if we use an AP with equal bandwidth (i.e., 8 Mbps), the downloading performance still drops due to offloading overhead, which includes AP association, DHCP, and connection re-establishment.

We report the SoftOffload decisions for all the testing cases in Table 5.9. As shown in the table, the context-aware decision made by SoftOffload does avoid using the 'weak' WiFi AP in the 4 Mbps case. For the 8 Mbps case, SoftOffload is able to make the correct decision by considering the overhead, as reflected in the Formula 5.1.

By experimenting with different file sizes, we also observe another fac-

Table 5.8: Download performance comparison benchmark.

| | Downloading Time (s) | | |
|---|---|---|---|
| | 33 MB | 65 MB | 141 MB |
| Baseline (8 Mbps) | 34.333 | 65.526 | 141.486 |
| Offload to AP2 (4 Mbps) | 49.585 | 115.436 | 267.825 |
| Offload to AP2 (8 Mbps) | 36.284 | 68.393 | 145.432 |
| Offload to AP2 (16 Mbps) | 34.377 | 49.732 | 88.456 |
| Offload to AP2 (24 Mbps) | 32.59 | 43.834 | 70.610 |

Table 5.9: Offloading decisions of SoftOffload.

| AP2 Bandwidth | Downloading Time (s) | | |
|---|---|---|---|
| | 33 MB | 65 MB | 141 MB |
| 4 Mbps | no offload | no offload | no offload |
| 8 Mbps | no offload | no offload | no offload |
| 16 Mbps | offload to AP2 | offload to AP2 | offload to AP2 |
| 24 Mbps | offload to AP2 | offload to AP2 | offload to AP2 |

tor, the amount of data, which can affect the offloading performance. As shown in the 16 Mbps case, even if we choose an AP with two times of bandwidth higher than the previous AP, the time for downloading a 33 MB file still increases (i.e., from 34.333 s to 34.377 s). This is a combined effect resulting from both the offloading overhead and the amount of data to offload. Although SoftOffload already considers the switching overhead and bandwidth context, it is difficult to predict future traffic amount generated by mobile devices. However, as shown in the table for the cases of 65 MB and 141 MB, if clients continue to offload traffic using APs with higher bandwidth, the performance will improve.

To highlight the effectiveness of SoftOffload, we further plot the experiment results in Figure 5.34 comparing the effects of offloading decisions. The 'Sub-optimal offloading' bars represent the results when a mobile device is connected to a congested AP which has only half of the bandwidth (4 Mbps) compared with the previously associated AP. The 'Effective offloading' bars show the results by following SoftOffload's suggestion to select the best available AP (24 Mbps in our case). As shown in Figure 5.34, there is a clear advantage to use SoftOffload to guide WiFi offload when the data volume is large. Meanwhile, when the offloading volume is relatively small, the improvement is less visible due to the overhead introduced by switching between APs.

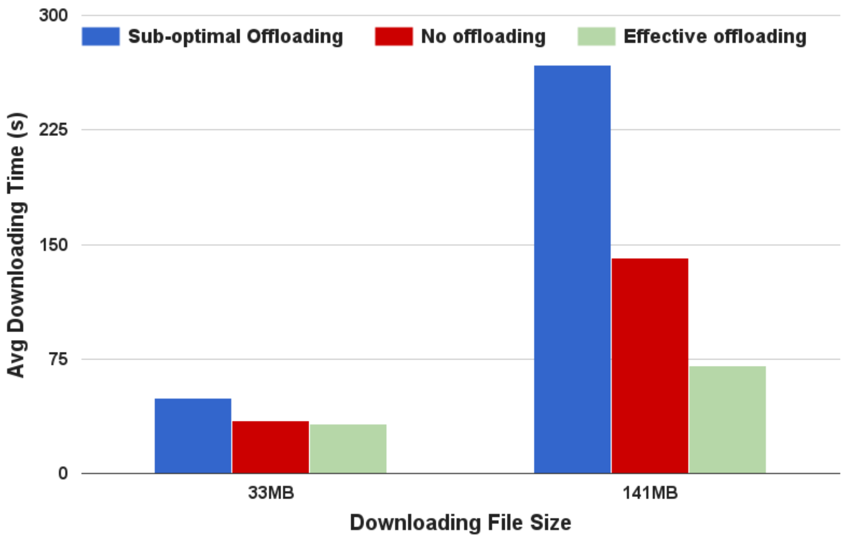


Figure 5.34: Context-aware offloading in static scenario [28].

As illustrated in our measurement study (Section 5.2.1), signal strength has a direct impact on the performance of WiFi offloading. This is reflected in the proposed context-aware formula where the offloading metric for choosing candidate AP is affected by two major factors:

$$(1 - e^{-k_0(S-k_1)}) \cdot \frac{estimatedBw}{\arg\max estimatedBw} \cdot \frac{restBw}{totalBw}$$

where $S$ is the signal level (RSSI) collected from the client side, and $Bw$ is the bandwidth parameter of a candidate AP. We set $k_0 = \frac{1}{3}$, and $k_1 = -73$ based on our measurement insight (Section 5.2.1).

To demonstrate SoftOffload can synthesize all crucial context, we evaluate the cases when both bandwidth and signal strength are considered. With a baseline bandwidth of 8 Mbps, we set up one WiFi AP as the target one and apply two bandwidth settings on it, including 8 Mbps and 16 Mbps. We repeat the downloading tests by placing the target AP to several dedicated spots away from the testing client, in order to adjust the user perceived signal strength. We report both metric values generated by SoftOffload and the throughput results obtained from live measurements in Table 5.10. The trend is clear in the table: when the signal strength and bandwidth decreases, our context-aware offloading formula yields the values in accordance with the trend. For example, when an AP is of weak signal strength -78 dBm (e.g., far away from the client), even if its nominal bandwidth is good (16 Mbps in this case), the overall performance is still poor, which matches to our offloading metric value -4.294. This indicates that SoftOffload will not select this AP as the offloading target, thereby avoiding the potential poor performance and energy cost.

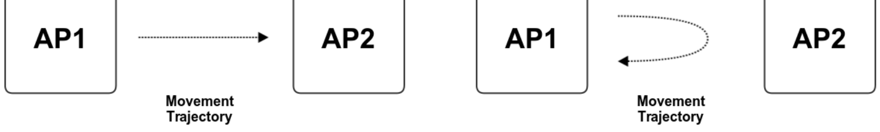
Table 5.10: Offloading metric values for different test cases.

| | -34 dBm 16 Mbps | -66 dBm 16 Mbps | -31 dBm 8 Mbps | -65 dBm 8 Mbps | -75 dBm 16 Mbps | -78 dBm 16 Mbps |
|---|---|---|---|---|---|---|
| Throughput (Mbps) | 15.12 | 15.04 | 7.84 | 7.52 | 3.68 | **3.04** |
| Metric | 0.999 | 0.903 | 0.5 | 0.465 | -0.947 | **-4.294** |

***Movement Estimation***

SoftOffload uses a short-term movement estimation scheme to assist its offloading decision. The goal is to adjust the offloading metric so that SoftOffload can recommend the best available AP to clients that are on the move for offloading their mobile data traffic. The overall movement estimation is enabled by the estimation factor $\mu \cdot (1 - e^{-k_0(S-k_1)})$ as part of the context-aware offloading formula. The central controller collects the scanning results (3 consecutive scanning) from a client and then calculates the value of $\mu$ as described in Section 5.4.1. SoftOffload will use the signal strength RSSI values from the last scanning as $S$ to calculate $(1 - e^{-k_0(S-k_1)})$.

To validate the estimation scheme, we set up two WiFi APs around 5 meters from each other using the network topology presented in Figure 5.33. This setting allows us to test two typical client movement trajectories as shown in Figure 5.35.



(a) Straight line client trajectory to AP2 (b) U-turn client trajectory back to AP1

Figure 5.35: Movement trajectories.

Table 5.11: Mobility prediction evaluation.

| | Accuracy of Movement Estimation for $\mu$ | Overall Accuracy of Movement and Signal Strength |
|---|---|---|
| Straight | 60% | 90% |
| U-Turn | 50% | 90% |

To focus on the movement effects, we configure both APs with the same setting, including bandwidth and WiFi channel. For the straight line trajectory, the testing client (Galaxy Tab 2) first connects to AP1 and generates data traffic to trigger the offloading process. When SoftOffload starts the offloading, we carry the tablet and walk toward AP2. For the U-turn trajectory, we first walk toward AP2 and turn back to AP1. The turning point is approximately 2 meters to AP2.

We repeat each movement scenario 10 times and summarize the estimation accuracy in Table 5.11 based on our measurement data. As shown in the table, there estimation for $\mu$ and overall estimation that considers both movement and signal strength. Regarding the accuracy of $\mu$ estimation, we only consider an estimation accurate if it exactly matches our movement pattern. For example, for the case of straight line trajectory, an estimation result is accurate only if it indicates that we are approaching AP2 and moving away from AP1 (i.e., $\mu_{AP2} = 1$ and $\mu_{AP1} = 0.7$). All other results are considered inaccurate. Regarding the overall estimation, we evaluate the metric value produced by $\mu \cdot (1 - e^{-k_0(S-k_1)})$. The estimation is considered accurate if the metric value guides the client to the correct AP. Take the U-turn case for example, the estimation is accurate if the metric value for AP1 is larger than that of AP2.

As shown in Table 5.11, the movement estimation for $\mu$ does not perform well in the live environment in both cases. This is mainly due to the fluctuation of RSSI values collected by the consecutive scanning. However, when we consider both movement and signal strength, the overall estimation accuracy improves to 90%. This indicates that the short-term mobility estimation can assist SoftOffload to make correct offloading decisions even for users on the move.

### *Overhead*

As an SDN-based solution, SoftOffload uses OpenFlow protocol to monitor switches, which is conducted periodically using a fixed interval. To understand the overhead of an SDN-based offloading solution, we plot the CPU and memory utilization in Figure 5.36 and Figure 5.37, respectively,

Table 5.12: Client-side overhead.

| | WiFi Scanning (3 times) | | Network Switching | |
|---|---|---|---|---|
| | Avg. Time (s) | Average Energy Consum. (uAh) | Avg. Time (s) | Average Energy Consum. (uAh) |
| Nexus S | 4.58 | 84.63 | 2.004 | 94.88 |
| Galaxy S2 | 14.66 | 356.21 | 1.35 | 180.19 |
| Galaxy S3 | 13.56 | 235.97 | 1.01 | 62.54 |
| Galaxy S4 | 13.23 | 299.67 | 1.22 | 83.55 |
| Galaxy S5 | 13.04 | 454.91 | 1.04 | 77.79 |

for running the central controller on a commodity hardware (desktop with Intel Core Duo E8500 CPU, 6GB RAM, Ubuntu 14.04 LTS, Linux kernel version 3.13.32). We adjust the monitoring interval to cover four cases: 5 seconds, 2 seconds, 1 second, and 0.5 second. The baseline shows the CPU and memory utilization when no controller is running. The testbed is shown in Figure 5.33 in which a mobile client downloads a large data file (300+ MB) over HTTP from the local file server. As shown in the figures, the overhead for running SoftOffload controller for monitoring the testbed network is roughly 250 MB and with relatively low CPU usage.

Because SoftOffload requires the collaboration of clients to enable context-aware offloading, it introduces extra overhead to the client-side, such as WiFi scanning and network switching. To illustrate the client-side overhead, we conduct a set of measurements on different smartphone models running the SoftOffload client application.

Figure 5.38 shows the experiment setup for recording the energy consumption in WiFi scanning and network switching. For comprehensiveness, we evaluate five smartphones models: Nexus S, Galaxy S2, S3, S4 and S5. In the experiment, we instruct SoftOffload controller to send commands to trigger WiFi scanning and network switching. We repeat each test case 10 times for all the smartphone models and report the mean values in Table 5.12.

As shown in the table, switching to a different WiFi network takes less than 2 seconds on average for most of the devices we measured. Except for Galaxy S2, the energy consumption needed for switching to another AP is below 100 uAh for all other devices. Given that the battery capacity on a modern Android smartphone is around 2500 mAh and energy consumption for network switching can be compared to the cost when the screen is turned on in standby mode for few seconds, we consider the overhead introduced by network switching reasonable.

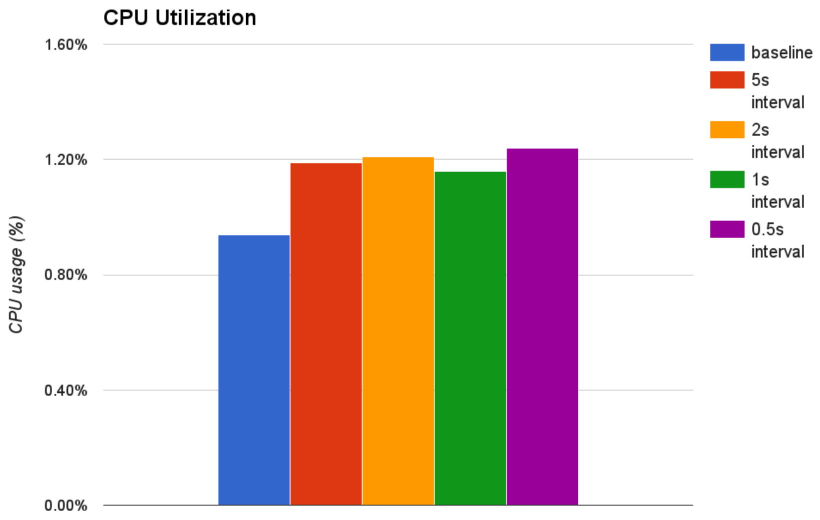


Figure 5.36: CPU usage of SoftOffload controller.

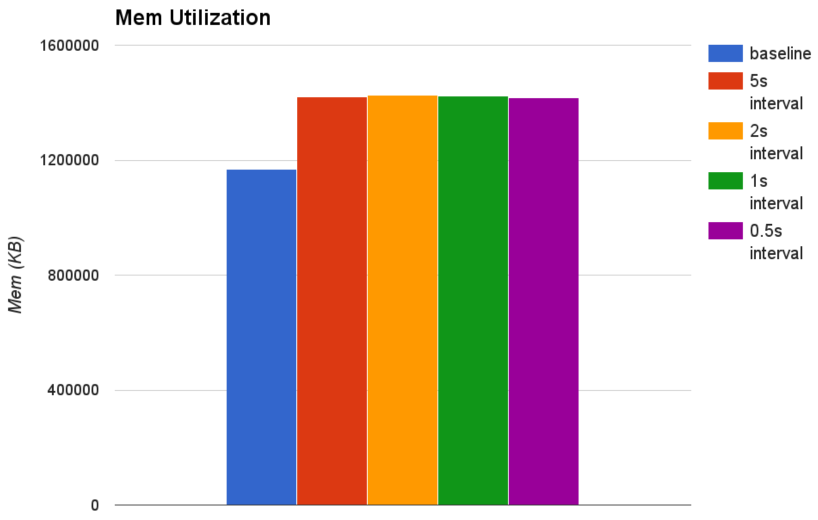


Figure 5.37: RAM usage of SoftOffload controller.

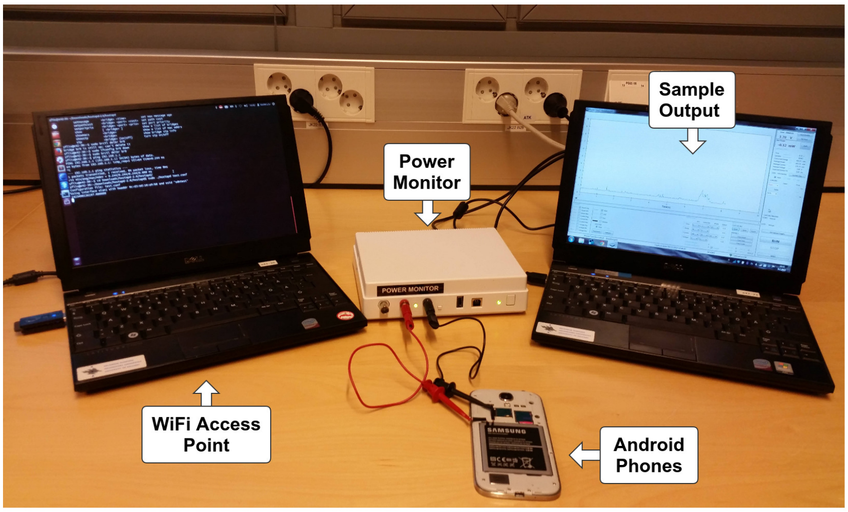


Figure 5.38: Overhead measurement setup.

Much to our surprise, the WiFi scanning, an important component to enable context-aware offloading, requires more than 13 seconds on several new Galaxy devices, including S2, S3, S4 and S5. As shown in the table, this behavior also results in extra energy cost on those devices. This observation indicates that our mobility estimation technique may introduce unreasonable overhead. Although such scanning behavior can be possibly fixed by hacking Android underlying drivers, a more practical approach is to adopt other mechanism such as using mobile sensors to help [282].

## 5.5 Discussion

In this section, we discuss potential deployment scenario and latest SDN-based solutions for WiFi offloading.

#### *Deployment Scenario*

To augment mobile traffic offloading, one visible trend is for MNOs to utilize third party WiFi services such as FON [13] to extend their carrier-class WiFi accesses. Given another fast growing trend favored by cloud service providers (e.g., Google and Amazon) that are interested in becoming mobile virtual network operators (MVNO), a potential deployment scenario of

[13] `https://corp.fon.com/en`

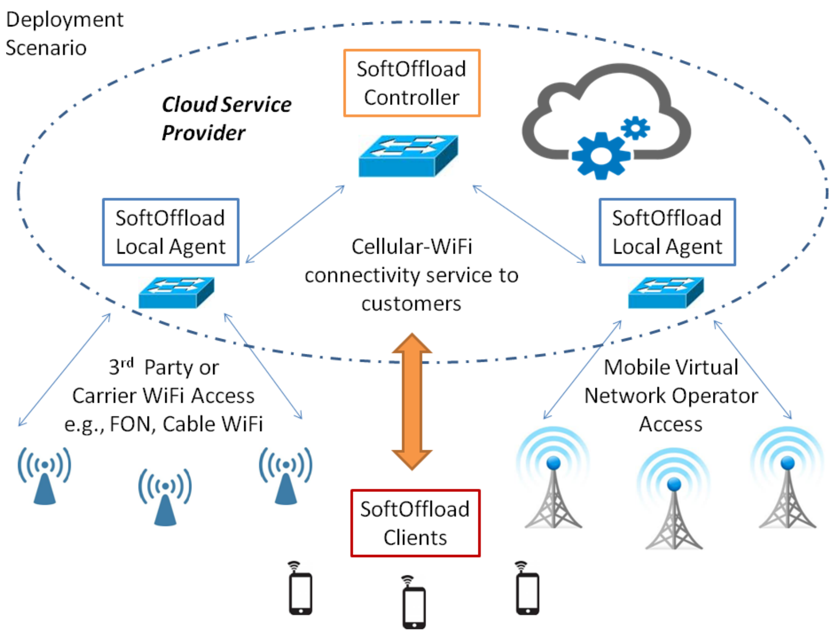

Deployment Scenario
Cloud Service Provider
SoftOffload Controller
SoftOffload Local Agent
Cellular-WiFi connectivity service to customers
SoftOffload Local Agent
3rd Party or Carrier WiFi Access e.g., FON, Cable WiFi
Mobile Virtual Network Operator Access
SoftOffload Clients

Figure 5.39: Deployment case for cloud service providers [28].

SoftOffload is to support cloud service providers to expanded their business by offering collaborative cellular-WiFi connectivity services.

Figure 5.39 shows a visioned view of unified cellular-WiFi access empowered by SoftOffload for cloud service providers. Through the collaborative design, non-intrusive QoE measurements and contextual information from end devices can be combined with the network-derived QoS insights to guide the end users always connecting to the best-performing access network.

Through the SoftOffload client extension (e.g., a dedicated application), end users are able to adjust parameters and select performance / cost profiles that best suit their individual needs. The local agents can function as a proxy at each access networks (e.g., WiFi or MVNO cellular access) to monitor and collect essential contextual information to support WiFi offloading. The central controller can be hosted by the cloud service provider to manage local agents deployed on the edge of cellular and WiFi accesses.

By using a flexible and extensible controlling framework to enable interworking of two types of access networks, cloud operators can adopt this approach to stand out in competition by offering a cellular-WiFi interworking package for their customers. The collaborative principle of SoftOffload makes it feasible to be adopted by cloud service providers and also used in the carrier-class WiFi scenario.

***SDN for WiFi Offloading***

Since SDN delivers a fine-grained control of network through its abstraction of the underlying hardware, it meets the urgent need from the mobile networks to simultaneously operate over multiple wireless technologies. A recent SDN-base proposal ATOM [283] provides an end-to-end traffic management solution that enables operators to manage traffic flows across heterogeneous LTE and WiFi networks. To enable cooperation of home WiFi APs, the COAP framework [284] offers an open APIs to proactively configure home APs for interference mitigation and perform traffic aware adaptations at APs. For LTE and upcoming 5G networks, researchers have investigated the architecture design [285], mobile backhaul support [31, 286], and integration framework with the existing networks [287].

Compared with existing proposals, SoftOffload is an open-source solution extending the collaboration principle to cover end user, WiFi providers and cellular operators which opens up deployment opportunities for new players in this domain (i.e., cloud service providers). Because it is hard to predict what new requirements will impose on mobile networks given the fast advance of communication and application, an open and extensible approach has more prospects than a rigid design to meet the unforeseen needs.

## 5.6 Summary

We present a novel software defined platform for achieving collaborative and context-aware traffic offloading. The platform consists of an extensible central controller, programmable offloading agents, and offloading extensions for mobile devices. By enabling collaboration between wireless networks and mobile users, our solution can make optimal offloading decisions that improve offloading efficiency for network operators and achieve energy saving for mobile users.

In this chapter, we presented SoftOffload, an open-source collaborative platform to augment WiFi offloading. Driven by our extensive measurements on smartphone energy consumption and performance for wireless communications, we propose an effective context-aware offloading algorithm and integrate it to our platform. We advocate that openness and extensibility are indispensable features for smooth deployment by allowing new services to be added over time for requirements of the fast evolving mobile networks. In this regard, SoftOffload offers an open and extensible framework to encourage collaboration and potential deployment by mobile carriers and cloud service providers.

The key contributions in SoftOffload include:

- We conduct a series of measurements exploring performance and energy consumption in WiFi offloading, which drives the design of SoftOffload.
- We exploit context awareness to enhance the gain of WiFi offloading as a use case for SoftOffload and propose a novel context-aware offloading algorithm.
- We develop a prototype system for SoftOffload and evaluate it in live networks. Our work not only explores the feasibility of context-based traffic offloading, but also provides guidelines for designing and implementing a centralized SDN platform for managing wireless networks.
- We highlight the opportunities for cloud service providers to utilize SoftOffload through the mobile virtual network operators (MVNO) channel and alliance with WiFi service providers.
- To encourage collaboration, we have released SoftOffload under open-source licenses. SoftOffload can be accessed via `http://www.cs.helsinki.fi/group/eit-sdn/softoffload.html`.

Note that, our goal is to augment WiFi offloading with a programmable and collaborative approach. We do not expect our solution to replace existing mechanisms from SDOs (e.g., 3GPP, IETF) or research communities. Instead, we provide an open and extensible platform to integrate existing techniques and promote development of novel network applications to meet the requirements of the fast evolving mobile networks.

In brief, the beauty of SoftOffload comes from its programmable and open design which improves the extensibility and deployability. SoftOffload facilitates the adding of new extensions catering for new requirements from the evolving access environment. Combining with SDN, our open-source oriented approach aims to bring community contributions to drive the development of SoftOffload even further.

# Chapter 6

# Conclusions

This chapter concludes the dissertation by *1)* summarizing the research contributions, *2)* discussing open issues and future work, and *3)* presenting concluding remarks.

## 6.1 Contributions

This dissertation presents the work on collaborative traffic offloading for mobile systems. We first investigate the concept of mobile traffic offloading and identify its key benefits, technological challenges, requirements, and current research directions in order to shed light on the impact and implications of traffic offloading for mobile users and network operators. The dissertation presents an extensive literature review and analysis for mobile traffic offloading (Section 2.2), mobile energy awareness (Section 2.3), and SDN applications for mobile environments (Section 2.4). These studies provide valuable input for the future design and development of mobile traffic offloading solutions.

Based on experimental investigations in operational networks and testbed, this dissertation shows how to enable the host and network driven traffic offloading through the NAO framework. We present measurement studies (Section 3.1), framework design (Section 3.2), system implementations (Section 3.3) and a comparison study of standard solutions (Section 3.4). By verifying the feasibility of NAO through live experiments in the operator networks, we demonstrate how to achieve traffic offloading in practice using the IETF standard compliant protocols. The value of NAO is on the system design and reference implementation, which is the first step towards collaborative traffic offloading. The observations and lessons from NAO provide insights for the follow-up work in this domain.

The MADNet proposal focuses on improving the energy efficiency for smartphones. Through measurement studies, this dissertation provides a thorough analysis on WiFi-based offloading by covering the metropolitan WiFi access, mobile scenarios, and energy cost (Section 4.2). The observations motivate the design of the MADNet energy-aware offloading, which harvests the collaboration across cellular operators, WiFi providers, and end users to improve the energy awareness of offloading for smartphones. The core of MADNet is the energy-aware algorithm and optimization techniques (Section 4.3). Based on prototype implementation and experimentation in live networks, we show that with dedicated optimization we are able to achieve reasonable energy saving in WiFi offloading (Section 4.4). The work in MADNet highlights the importance of collaboration and reveals the factors that were overlooked by previous work. The findings deepen our understanding on the energy aspect of mobile traffic offloading and further show how to enable energy awareness in practice.

Finally, based on the lessons from NAO and MADNet, this dissertation proposes SoftOffload to enhance the openness and extensibility for expanding the solution deployment. Through measurement studies, we identify the key factors in WiFi-based offloading and show how they affect the performance and energy consumption (Section 5.2). Built on the SDN paradigm, the centralized architecture enables the context-aware algorithm to collaboratively collect context information for guiding WiFi offloading (Section 5.3). The implementation and experimentation also reveal the benefits and overhead of an SDN-based solution (Section 5.4). Being a novel open-source platform for traffic offloading, SoftOffload adopts open standard SDN solutions and is released under open-source licenses to encourage contributions from the community. In addition, this dissertation highlights the opportunities that cloud service providers can utilize SoftOffload through the mobile virtual network operators (MVNO) channel and alliance with WiFi service providers (Section 5.5). The work in SoftOffload demonstrates how to transfer a static and closed mobile traffic offloading solution into a dynamic and open one. The platform itself is a valuable asset for the community to experiment and further add extensions catering for the new requirements of the evolving mobile access environment.

## 6.2 Open Issues and Future Work

One of the key questions in mobile traffic offloading concerns the role of the network provider and end users in the offloading process (i.e., who should drive the offloading and make the decision?) In this regard, an operator may have a better knowledge for network resources and conditions while an end user only has a partial and local view of the access network. On the other hand, an operator-driven strategy often tends to give priority to a certain type of traffic or class of users, while a distributed offloading strategy driven by users may avoid such fairness issue. Since both strategies have advantages and disadvantages of its own. The debate on this issue is still open, and we believe more research is needed in addition to the work presented in this dissertation.

Regarding user adoption, we highlight that user collaboration is a key enabler for any offloading solution. The central question is how to motivate users to participate. Mobile operators who intend to offload mobile data traffic are responsible to propose a business concept to motivate their customers and make offloading attractive and fully functional. Although there is existing work [9] exploring the incentive scheme, additional issues still remain, including security and privacy involved in such collaborative offloading process. This is a potential topic we intend to explore in the future.

Another technical challenge is the lack of a standardized and widely accepted mechanism to handle several flows in parallel on different interfaces. As pointed out in our literature review, various mechanisms have been proposed [58, 178], but we do not yet reach a consensus. From the user perspective, a major concern comes from the dramatic battery drain to operate multiple wireless interfaces simultaneously. For operators, there is a privacy concern to prevent them to force a mobile device to turn on and off network interfaces. We consider possible solutions to explore low-powered network interfaces and energy saving design as our future work to augment mobile traffic offloading. We also plan to investigate how to leverage the collaborative context awareness enabled by our platform to optimize mobile sensing activities (e.g., scanning interval and sampling rate) in different environments. In addition, our experimental research covers mainly single user point of view and we intend to investigate the scalability aspect of our proposals in the future.

An interesting future research topic concerns how to merge the various and often stand-alone offloading proposals in a fully integrated network architecture. This unified design requires reconsidering existing wireless network paradigms. In this regard, the proposed SoftOffload is our first step to explore an extensible platform to facilitate integration and development of offloading schemes. Since we released the platform in early 2015, we plan to enhance the north-bound APIs for developers to utilize SoftOffload and further extend its functionality.

## 6.3 Concluding Remarks

One of the most engaging challenges for mobile network operators today is how to manage the exponential data traffic increase. Mobile traffic offloading stands out as a promising and low cost solution to reduce the burden on the mobile networks.

In this dissertation, we review the technical aspects and discuss the state of the art in this domain. We describe and present a set of novel design and functionalities needed to enable collaborative traffic offloading. Regarding the promising SDN-based traffic offloading as demonstrated in SoftOffload, we explore the open-source oriented approach and provide guidelines for designing and implementing a centralized SDN platform to manage multi-access wireless networks.

Mobile service providers are nowadays challenged by facing competition not only from other carriers but also from over-the-top Internet services, such as Google, Amazon, and Facebook, who are interested in becoming mobile virtual network operators. We believe that in the upcoming mobile networks such as 5G, mobile traffic offloading is extremely important because it enables mobile operators to optimize RAN resources, and improve the quality of experience (QoE) for data-intensive mobile applications. In addition, an open-source collaborative solution such as SoftOffload can open new opportunities to bridge the gap between cellular and WiFi services providers.

# References

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019 White Paper. February 3, 2015. Available at: `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html`. Visited on 07.07.2015.

[2] Cisco Visual Networking Index: Forecast and Methodology, 2014–2019. May 27, 2015. Available at: `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html`. Visited on 07.07.2015.

[3] A. Aijaz, H. Aghvami, M. Amani, "A Survey on Mobile Data Offloading: Technical and Business Perspectives". In *IEEE Wireless Communications*, IEEE, 2013, Vol. 20, No. 2, pp. 104–112.

[4] K. Samdanis, T. Taleb, S. Schmid, "Traffic Offload Enhancements for eUTRAN". In *IEEE Communications Surveys & Tutorials*, IEEE, 2013, Vol. 14, No. 3, pp. 884–896.

[5] A. Balasubramanian, R. Mahajan, A. Venkataramani, "Augmenting Mobile 3G Using WiFi". In *Proceedings of the 8th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, ACM, 2010, pp. 209–222.

[6] K. Lee, et al., "Mobile Data Offloading: How Much Can WiFi Deliver?" In *Proceedings of the 6th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT '10)*, ACM, 2010, Article 26, 12 pages.

[7] U. Shevade, et al., "Enabling Enabling High-Bandwidth Vehicular Content Distribution". In *Proceedings of the 6th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT '10)*, ACM, 2010, Article 23, 12 papges.

[8] X. Hou, P. Deshpande, S. R. Das, "Moving Bits from 3G to Metro-Scale WiFi for Vehicular Network Access: An Integrated Transport Layer Solution". In *Proceedings of the 19th IEEE International Conference on Network Protocols (ICNP '11)*, IEEE, 2011, pp. 353–362.

[9] X. Zhuo, W. Gao, G. Cao, Y. Dai, "Win-Coupon: An incentive framework for 3G traffic offloading". In *Proceedings of the 19th IEEE International Conference on Network Protocols (ICNP '11)*, IEEE, 2011, pp. 206–215.

[10] S. Dimatteo, P. Hui, B. Han, V. Li, "Cellular Traffic Offloading through WiFi Networks". In *Proceedings of the 8th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS '11)*, IEEE, 2011, pp. 192–201.

[11] N. Ristanovic, J. Le Boudec, A. Chaintreau, V. Erramilli, "Energy Efficient Offloading of 3G Networks". In *Proceedings of the 8th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS '11)*, IEEE, 2011, pp. 202–211.

[12] E. Bulut, B. K. Szymanski, "WiFi Access Point Deployment for Efficient Mobile Data Offloading". In *Proceedings of the 1st ACM MobiCom Workshop on Practical Issues and Applications in Next Generation Wireless Networks (PINGEN '12)*, ACM, 2012, pp. 45–50.

[13] A. Y. Ding, J. Crowcroft, "Use Case: Energy Awareness in Mobile Traffic Offloading". In *Smartphone Energy Consumption: Modelling and Optimization*, Cambridge University Press, 2014, Ch. 13, Sec. 6.

[14] J. Zhang, G. de la Roche, *Femtocells: Technologies and Deployment*, Wiley Publishing, 2010.

[15] S. Liu, and A. Striegel, "Casting Doubts on the Viability of WiFi Offloading". In *Proceedings of the ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design (CellNet '12)*, ACM, 2012, pp. 25–30.

[16] A. Y. Ding, B. Han, Y. Xiao, P. Hui, A. Srinivasan, M. Kojo, S. Tarkoma, "Enabling Energy-Aware Collaborative Mobile Data Offloading for Smartphones". In *Proceedings of the 10th Annual IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '13)*, IEEE, 2013, pp. 487–495.

[17] A. Y. Ding, J. Korhonen, T. Savolainen, M. Kojo, J. Ott, S. Tarkoma, and J. Crowcroft, "Bridging the Gap between Internet Standardization and Networking Research". In *ACM SIGCOMM Computer Communication Review*, ACM, 2014, Vol. 44, No. 1, pp. 56–62.

[18] A. Y. Ding, J. Korhonen, T. Savolainen, M. Kojo, S. Tarkoma, J. Crowcroft, "Bridge Networking Research and Internet Standardization: Case Study on Mobile Traffic Offloading and IPv6 Transition Technologies". In *IAB Workshop on Internet Technology Adoption and Transition (ITAT '13)*, Internet Architecture Board, 2013, 10 pages.

[19] S. Tarkoma, M. Siekkinen, E. Lagerspetz, Y. Xiao, *Smartphone Energy Consumption: Modelling and Optimization*, Cambridge University Press, Cambridge, 2014.

[20] I. Järvinen, B. Chemmagate, A. Y. Ding, L. Daniel, M. Isomäki, J. Korhonen, M. Kojo, "Effect of Competing TCP Traffic on Interactive Real-Time Communication". In *Proceedings of the 14th International Conference on Passive and Active Measurement (PAM '13)*, Springer-Verlag, 2013, pp. 94–103.

[21] I. Järvinen, Y. Ding, A. Nyrhinen, M. Kojo, "Harsh RED: Improving RED for Limited Aggregate Traffic". In *Proceedings of the 2012 IEEE 26th International Conference on Advanced Information Networking and Applications (AINA '12)*, IEEE, 2012, pp. 832–840.

[22] I. Järvinen, B. Chemmagate, A. Y. Ding, L. Daniel, M. Kojo, "Experimentation with Wireless Broadband Traffic Behaviour in WiBrA Project". Department of Computer Science, University of Helsinki, Technical Report, Series of Publications C-2012-19, 2012, 42 pages.

[23] Y. Ding, T. Savolainen, J. Korhonen, S. Tarkoma, P. Hui, M. Kojo, "NAO: A Framework to Enable Efficient Mobile Offloading". In *Proceedings of the ACM Middleware Workshop on Posters and Demos Track (PDT '11)*, ACM, 2011, Article 8, 2 pages.

[24] J. Korhonen, T. Savolainen, Y. Ding, "Controlling Traffic Offloading Using Neighbor Discovery Protocol". Internet-Draft "draft-korhonen-mif-ra-offload-03.txt", work-in-progress, IETF Secretariat, October 2011.

[25] J. Korhonen, T. Savolainen, A. Y. Ding, "Controlling Traffic Offloading Using Neighbor Discovery Protocol". Internet-Draft "draft-korhonen-mif-ra-offload-05.txt", work-in-progress, IETF Secretariat, August 2012.

[26] J. Korhonen, T. Savolainen, A. Y. Ding, M. Kojo, "Toward Network Controlled IP Traffic Offloading". In *IEEE Communications Magazine*, IEEE, 2013, Vol. 51, No. 3, pp. 96–102.

[27] A. Y. Ding, P. Hui, M. Kojo, S. Tarkoma, "Enable Energy-Aware Mobile Data Offloading for Smartphones through Vertical Collaboration". In *Proceedings of the 2012 ACM Conference on CoNEXT (CoNEXT '12) Student Workshop*, ACM, 2012, pp. 27–28.

[28] A. Y. Ding, Y. Liu, S. Tarkoma, H. Flinck, H. Schulzrinne, J. Crowcroft, "Vision: Augmenting WiFi Offloading with An Open-source Collaborative Platform". In *Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services (MCS '15)*, ACM, 2015, pp. 44–48.

[29] A. Y. Ding, Y. Liu, S. Tarkoma, H. Flinck, J. Crowcroft, "Demo: An Open-source Software Defined Platform for Collaborative and Energy-aware WiFi Offloading". In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*, ACM, 2015, 182–184.

[30] A. Y. Ding, J. Crowcroft, S. Tarkoma, "Poster: SoftOffload: A Programmable Approach Toward Collaborative Mobile Traffic Offloading". In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '14)*, ACM, 2014, pp. 368–368.

[31] J. Costa-Requena, R. Kantola, J. Llorente, V. Ferrer, J. Manner, A. Y. Ding, Y. Liu, S. Tarkoma, "Software Defined 5G Mobile Backhaul". In *Proceedings of the 1st International Conference on 5G for Ubiquitous Connectivity (5GU '14)*, IEEE, 2014, pp. 258–263.

[32] A. Y. Ding, J. Crowcroft, S. Tarkoma, H. Flinck, "Software Defined Networking for Security Enhancement in Wireless Mobile Networks". In *Computer Networks*, Elsevier, 2014, Vol. 66, pp. 94–101.

[33] L. Suomalainen, E. Nikkhouy, A. Y. Ding, S. Tarkoma, "Open Source Platforms, Applications and Tools for Software-Defined Networking and 5G Research". Department of Computer Science, University of

Helsinki, Technical Report, Series of Publications C-2014-2, 2014, 33 pages.

[34] Y. Liu, A. Y. Ding, S. Tarkoma, "Software-Defined Networking in Mobile Access Networks". Department of Computer Science, University of Helsinki, Technical Report, Series of Publications C-2013-1, 2013, 29 pages.

[35] 802.16e-2005 - IEEE Standard for Local and Metropolitan Area Networks - Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems - Amendment for Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands. IEEE.

[36] 802.16-2009 IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Broadband Wireless Access Systems. IEEE.

[37] M.2012: Detailed specifications of the terrestrial radio interfaces of International Mobile Telecommunications Advanced (IMT-Advanced). ITU-R.

[38] 802.16m-2011 - IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Broadband Wireless Access Systems Amendment 3: Advanced Air Interface. IEEE.

[39] TS 36.300 Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2. 3GPP.

[40] TS 23.401 General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access (Release 8). 3GPP.

[41] TS 24.301 Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3. 3GPP.

[42] M.687: International Mobile Telecommunications-2000 (IMT-2000). ITU-R.

[43] TR 37.976 Measurement of radiated performance for Multiple Input Multiple Output (MIMO) and multi-antenna reception for High Speed Packet Access (HSPA) and LTE terminals (Release 11). 3GPP.

[44] Long Term HSPA Evolution Mobile broadband evolution beyond 3GPP Release 10. Nokia Networks.

[45] A. J. Viterbi, *CDMA: Principles of Spread Spectrum Communication*, Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1995.

[46] CDMA2000 High Rate Packet Data Air Interface Specification. 3GPP2.

[47] DO Advanced: Maximizing the Performance of EV-DO. Qualcomm.

[48] IEEE Standards Association. "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems- Local and Metropolitan Area Networks- Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". ANSI/IEEE Std 802.11, 1999 Edition, 1999.

[49] IEEE Standards Association. "802.11b-1999 - IEEE Standard for Information Technology - Telecommunications and information exchange between systems - Local and Metropolitan networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher Speed Physical Layer (PHY) Extension in the 2.4 GHz band". IEEE, 1999.

[50] IEEE Standards Association. "802.11a-1999 - IEEE Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: High Speed Physical Layer in the 5 GHz band". IEEE, 1999.

[51] IEEE Standards Association. "802.11g-2003 - IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Further Higher Data Rate Extension in the 2.4 GHz Band". IEEE, 2003.

[52] IEEE Standards Association. "802.11-2012 - IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Clause 20". IEEE, 2012.

[53] IEEE Standards Association. "802.11ac-2013 - IEEE Standard for Information technology –Telecommunications and information exchange between systems - Local and metropolitan area networks–

Specific requirements–Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications–Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz". IEEE, 2013.

[54] IEEE Standards Association. "802.11ad-2012 - IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band". IEEE, 2012.

[55] M. Stemm, R. H. Katz, "Vertical Handoffs in Wireless Overlay Networks". In *Mobile Networks and Applications*, Springer-Verlag, 1998, Vol. 3, No. 4, pp. 335–350.

[56] TS 23.402 Architecture enhancements for non-3GPP accesses (Release 8). 3GPP.

[57] TR 23.882 3GPP system architecture evolution (SAE): Report on technical options and conclusions. 3GPP.

[58] TS 24.302 Access to the 3GPP Evolved Packet Core (EPC) via Non-3GPP Access Networks; Stage 3. 3GPP.

[59] R. Jain, S. Paul, "Network virtualization and software defined networking for cloud computing: a survey". In *IEEE Communications Magazine*, IEEE, 2013, Vol. 51, No. 11, pp. 24–31.

[60] Core Specification. Bluetooth Special Interest Group.

[61] I. F. Akyildiz, J. Xie, S. Mohanty, "A survey of mobility management in next-generation all-IP-based wireless systems". In *IEEE Wireless Communications*, IEEE, 2004, Vol. 11, No. 4, pp. 16–28.

[62] J. Korhonen, "IP mobility in wireless operator networks". Department of Computer Science, University of Helsinki, Doctoral Thesis, Series of Publications A-2008-4, 2008.

[63] K. Pahlavan, P. Krishnamurthy, A. Hatami, M. Ylianttila, J. Mäkelä, R. Pichna, J. Vallstron, "Handoff in hybrid mobile data networks". In *Personal Communications*, IEEE, 2000, Vol. 7, No. 2, pp. 34–47.

[64] J. Mäkelä, M. Ylianttila, K. Pahlavan, "Handoff decision in multi-service networks". In *Proceedings of the 11th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '00)*, IEEE, 2000, pp. 655–659.

[65] M. Ylianttila, M. Pande, J. Mäkelä, P. Mahonen, "Optimization scheme for mobile users performing vertical handoffs between IEEE 802.11 and GPRS/EDGE networks". In *Proceedings of the Global Telecommunications Conference (GLOBECOM '01)*, IEEE, 2001, pp. 3439–3443.

[66] M. Ylianttila, J. Mäkelä, P. Mahonen, "Supporting resource allocation with vertical handoffs in multiple radio network environment". In *Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '02)*, IEEE, 2002, pp. 64–68.

[67] M. Ylianttila, J. Mäkelä, K. Pahlavan, "Analysis of handoff in a location-aware vertical multi-access network". In *Computer Networks*, Elsevier, 2005, Vol. 47, No. 2, pp. 185–201.

[68] L. Daniel, M. Kojo, "Adapting TCP for Vertical Handoffs in Wireless Networks ". In *Proceedings of the 31st IEEE International Conference on Local Computer Networks (LCN '06)*, IEEE, 2006, pp. 151–158.

[69] L. Daniel, "Cross-layer Assisted TCP Algorithms for Vertical Handoff". Department of Computer Science, University of Helsinki, Doctoral Thesis, Series of Publications A-2010-6, 2010.

[70] C. Perkins, "IP Mobility Support for IPv4, Revised". Internet RFCs, ISSN 2070-1721, RFC 5944, 2010.

[71] C. Perkins, D. Johnson, J. Arkko, "Mobility Support in IPv6". Internet RFCs, ISSN 2070-1721, RFC 6275, 2011.

[72] K. Leung, G. Dommety, P. Yegani, K. Chowdhury, "WiMAX Forum / 3GPP2 Proxy Mobile IPv4". Internet RFCs, ISSN 2070-1721, RFC 5563, 2010.

[73] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, B. Patil, "Proxy Mobile IPv6". Internet RFCs, ISSN 2070-1721, RFC 5213, 2008.

[74] IEEE Standards Association. "802.21-2008 IEEE Standard for Local and metropolitan area networks - Media Independent Handover Services". IEEE, 2008.

[75] OpenFlow Specifications. Open Networking Foundation.

[76] R. Draves, D. Thaler, "Default Router Preferences and More-Specific Routes". Internet RFCs, ISSN 2070-1721, RFC 4191, 2005.

[77] P. Bahl, and V. N. Padmanabhan: "RADAR: An In-Building RF-Based User Location and Tracking System". In *Proceedings of the 19th IEEE International Conference on Computer Communications (INFOCOM '00)*, IEEE, 2000, pp. 775–784.

[78] R. Droms, "Dynamic Host Configuration Protocol" Internet RFCs, ISSN 2070-1721, RFC 2131, 1997.

[79] P. Mockapetris, "DOMAIN NAMES - CONCEPTS AND FACILITIES" Internet RFCs, ISSN 2070-1721, RFC 1034, 1987.

[80] P. Mockapetris, "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION" Internet RFCs, ISSN 2070-1721, RFC 1035, 1987.

[81] R. Chakravorty, et al., "Performance Issues with Vertical Handovers - Experiences from GPRS Cellular and WLAN Hot-spots Integration". In *Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications (PERCOM '04)*, IEEE, 2004, pp 155–164.

[82] M. Shin, A. Mishra and W. A. Arbaugh, "Improving the Latency of 802.11 Hand-offs using Neighbor Graphs". In *Proceedings of the 2nd ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '04)*, ACM, 2004, pp 70–83.

[83] V. Birk, A. Mishra, and S. Banerjee, "Eliminating Handoff Latencies in 802.11 WLANs using Multiple Radios: Applications, Experience, and Evaluation". In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement (IMC '05)*, USENIX Association, 2005, pp. 299–304.

[84] I. Ramani, and S. Savage, "SyncScan: Practical Fast Handoff for 802.11 Infrastructure Networks". In *Proceedings of the 24th IEEE International Conference on Computer Communications (INFOCOM '05)*, IEEE, 2005, pp. 675–684.

[85] P. Deshpande, A. Kashyap, C. Sung, and S. R. Das, "Predictive Methods for Improved Vehicular WiFi Access". In *Proceedings of the 7th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, ACM, 2009, pp. 263–276.

[86] A. J. Nicholson, and B. D. Noble, "BreadCrumbs: Forecasting Mobile Connectivity". In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking (MobiCom '08)*, ACM, 2008, pp. 46–57.

[87] A. J. Nicholson, et al., "Improved Access Point Selection". In *Proceedings of the 4th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '06)*, ACM, 2006, pp. 233–245.

[88] Core Specification 4.2. Bluetooth Special Interest Group.

[89] K. Fukuda, K. Nagami, "A Measurement of Mobile Traffic Offloading". In *Proceedings of the 14th Passive and Active Measurement Conference (PAM '13)*, Springer, 2013, pp. 73–82.

[90] R. R. Stewart, "Stream Control Transmission Protocol". Internet RFCs, ISSN 2070-1721, RFC 4960, 2007.

[91] B. Han, et al., "Cellular Traffic Offloading Through Opportunistic Communications: A Case Study". In *Proceedings of the 5th ACM Workshop on Challenged Networks (CHANTS '10)*, ACM, 2010, pp. 31–38.

[92] B. Han, P. Hui, A. Srinivasan, "Mobile data offloading in metropolitan area networks". In *SIGMOBILE Mobile Computing and Communications Review*, ACM, 2010, Vol. 14, No. 4, pp. 28–30.

[93] B. Han, et al., "Mobile Data Offloading through Opportunistic Communications and Social Participation". In *IEEE Transactions on Mobile Computing*, IEEE, 2012, Vol. 11, No. 5, pp. 821–834.

[94] X. Wang, et al., "TOSS: Traffic offloading by social network service-based opportunistic sharing in mobile social networks". In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM '14)*, IEEE, 2014, pp. 2346–2354.

[95] P. Baier, F. Durr, K. Rothermel, "TOMP: Opportunistic traffic offloading using movement predictions". In *Proceedings of the 37th Conference on Local Computer Networks (LCN '12)*, IEEE, 2012, pp. 50–58.

[96] F. Rebecchi, M. Dias de Amorim, V. Conan, “DROid: Adapting to individual mobility pays off in mobile data offloading”. In *Proceedings of IFIP Networking Conference*, IEEE, 2014, pp. 1–9.

[97] N. Vallina-Rodriguez, C. Efstratiou, G. Xie, J. Crowcroft, “Enabling Opportunistic Resources Sharing on Mobile Operating Systems: Benefits and Challenges”. In *Proceedings of the 3rd ACM Workshop on Wireless of the Students, by the Students, for the Students (S3 ’11)*, ACM, 2011, pp. 29–32.

[98] S. Liu, A. D. Striegel, “Exploring the Potential in Practice for Opportunistic Networks Amongst Smart Mobile Devices”. In *Proceedings of the 19th ACM International Conference on Mobile Computing and Networking (MobiCom ’13)*, ACM, 2013, pp. 315–326.

[99] J. Ott, D. Kutscher, “Drive-thru Internet: IEEE 802.11b for Automobile Users”. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM ’04)*, IEEE, 2004. Paper 122, 12 pages.

[100] J. Ott, D. Kutscher, “A Disconnection-tolerant Transport for Drive-thru Internet Environments”. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM ’05)*, IEEE, 2005. Paper 27, 14 pages.

[101] V. Bychkovsky, et al., “A Measurement Study of Vehicular Internet Access Using In Situ WiFi Networks”. In *Proceedings of the 12th ACM International Conference on Mobile Computing and Networking (MobiCom ’06)*, ACM, 2006, pp. 50–61.

[102] P. Deshpande, X. Hou, S. R. Das, “Performance Comparison of 3G and Metro-scale WiFi for Vehicular Network Access”. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC ’10)*, ACM, 2010, pp. 301–307.

[103] J. Eriksson, H. Balakrishnan, S. Madden, “Cabernet: Vehicular Content Delivery Using WiFi”. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking (MobiCom ’08)*, ACM, 2008, pp. 199–210.

[104] P. Rodriguez, et al., “MAR: A Commuter Router Infrastructure for the Mobile Internet”. In *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services (MobiSys ’04)*, ACM, 2004, pp. 217–230.

[105] C. Nicutar, D. Niculescu, C. Raiciu, "Using Cooperation for Low Power Low Latency Cellular Connectivity". In *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies (CoNEXT '14)*, ACM, 2014, pp. 337–348.

[106] K. Kumar, J. Liu, Y. Lu, B. Bhargava, "A Survey of Computation Offloading for Mobile Systems". In *Mobile Networks and Applications*, Springer, 2012, Vol. 18, No. 1, pp 129–140.

[107] E. Cuervo, et al., "MAUI: Making Smartphones Last Longer with Code Offload". In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, ACM, 2010, pp. 49–62.

[108] R. Kemp, N. Palmer, T. Kielmann, H. Bal, "Cuckoo: A Computation Offloading Framework for Smartphones". In *Proceedings of Second International ICST Conference on Mobile Computing, Applications, and Services (MobiCASE '10)*, Springer, 2010. pp. 59–79.

[109] B. Chun, et al., "CloneCloud: Elastic Execution Between Mobile Device and Cloud". In *Proceedings of the Sixth Conference on Computer Systems (EuroSys '11)*, ACM, 2011. pp. 301–314.

[110] S. Kosta, et al., "ThinkAir: Dynamic Resource Allocation and Parallel Execution in the Cloud for Mobile Code Offloading". In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM '12)*, IEEE, 2012. pp. 945–953.

[111] TR 23.861 Network Based IP Flow Mobility. 3GPP.

[112] TR 23.853 Operator Policies for IP Interface Selection (OPIIS). 3GPP.

[113] TR 23.829 Local IP Access and Selected IP Traffic Offload (LIPA-SIPTO). 3GPP.

[114] TS 23.261 IP Flow Mobility and Seamless Wireless Local Area Network (WLAN) Offload; Stage 2. 3GPP.

[115] H. Soliman, "Mobile IPv6 Support for Dual Stack Hosts and Routers". Internet RFCs, ISSN 2070-1721, RFC 5555, 2009.

[116] TR 23.852 Study on S2a Mobility based on GPRS Tunnelling Protocol (GTP) and Wireless Local Area Network (WLAN) access to the Enhanced Packet Core (EPC) network (SaMOG); Stage 2. 3GPP.

[117] TS 23.327 Mobility between 3GPP-Wireless Local Area Network (WLAN) interworking and 3GPP systems. 3GPP.

[118] F. Rebecchi, et al., "Data Offloading Techniques in Cellular Networks: A Survey". In *IEEE Communications Surveys and Tutorials*, IEEE, 2014.

[119] V. Chandrasekhar, J. G. Andrews, A. Gatherer, "Femtocell Networks: A Survey". In *IEEE Communications Magazine*, IEEE, 2008, Vol. 46, No. 9, pp. 59–57.

[120] A. Carroll, G. Heiser, "An Analysis of Power Consumption in a Smartphone". In *Proceedings of the USENIX Annual Technical Conference (ATC '10)*, USENIX Association, 2010.

[121] A. Rice, S. Hay, "Decomposing Power Measurements for Mobile Devices". In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom '10)*, IEEE, 2010, pp. 70–78.

[122] R. Trestian, et al., "Energy Consumption Analysis of Video Streaming to Android Mobile devices". In *Proceedings of the IEEE Network Operations and Management Symposium (NOMS '12)*, IEEE, 2012, pp. 444–452.

[123] S. Chandra, "Wireless Network Interface Energy Consumption: Implications for Popular Streaming Formats". In *Multimedia Systems*, Springer-Verlag, 2003, Vol. 9, No. 2, pp. 185–201.

[124] J. Yao, et al., "An Empirical Study of Bandwidth Predictability in Mobile Computing". In *Proceedings of the 3rd ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH '08)*, ACM, 2008, pp. 11–18.

[125] H. Falaki, et al., "A First Look at Traffic on Smartphones". In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC '10)*, ACM, 2010, pp. 281–287.

[126] G. Maier, F. Schneider, A. Feldmann, "A First Look at Mobile Handheld Device Traffic". In *Proceedings of the 11th Passive and Active Measurement Conference (PAM '10)*, Springer, 2010, pp. 161–170.

[127] H. Falaki, et al., "Diversity in Smartphone Usage". In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, ACM, 2010, pp. 179–194.

[128] A. Gember, A. Anand, A. Akella, “A Comparative Study of Handheld and Non-Handheld Traffic in Campus Wi-Fi Networks”. In *Proceedings of the 12th Passive and Active Measurement Conference (PAM ’11)*, Springer, 2011, pp. 173–183.

[129] M. Heikkinen, A. W. Berger, “Comparison of User Traffic Characteristics on Mobile-Access versus Fixed-Access Networks”. In *Proceedings of the 13th Passive and Active Measurement Conference (PAM ’12)*, Springer, 2012, pp. 32–41.

[130] N. Balasubramanian, et al., “Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications”. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement (IMC ’09)*, ACM, 2009, pp. 280–293.

[131] A. Auçinas, N. Vallina-Rodriguez, Y. Grunenberger, V. Erramili, K. Papagiannaki, J. Crowcroft, D. Whetheral, “Staying Online While Mobile: The Hidden Costs”. In *Proceedings of the 9th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT ’13)*, ACM, 2013, pp. 315–320.

[132] G. P. Perrucci, et al., “On the impact of 2G and 3G network usage for mobile phones’ battery life”. In *Proceedings of the European Wireless Conference (EW ’09)*, IEEE, 2009, pp. 255–259.

[133] J. Wigard, et al., “On the User Performance of LTE UE Power Savings Schemes with Discontinuous Reception in LTE”. In *Proceedings of the IEEE International Conference on Communications Workshops*, IEEE, 2009, pp. 1–5.

[134] F. Qian, et al., “Characterizing Radio Resource Allocation for 3G Networks”. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC ’10)*, ACM, 2010, pp. 137–150.

[135] J. Huang, et al., “A Close Examination of Performance and Power Characteristics of 4G LTE Networks”. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys ’12)*, ACM, 2012, pp. 225–238.

[136] S. Rosen, et al., “Discovering Fine-grained RRC State Dynamics and Performance Impacts in Cellular Networks”. In *Proceedings of the 20th ACM International Conference on Mobile Computing and Networking (MobiCom ’14)*, ACM, 2014, pp. 177–188.

[137] F. Qian, et al., "Periodic Transfers in Mobile Applications: Network-wide Origin, Impact, and Optimization". In *Proceedings of the 21st International Conference on World Wide Web (WWW '12)*, ACM, 2012, pp. 51–60.

[138] J. Huang, et al., "Screen-off Traffic Characterization and Optimization in 3G/4G Networks". In *Proceedings of the 12th ACM SIGCOMM Conference on Internet Measurement (IMC '12)*, ACM, 2012, pp. 357–364.

[139] X. He, et al., "A Panoramic View of 3G Data/Control-plane Traffic: Mobile Device Perspective". In *Proceedings of the 11th International IFIP TC 6 Conference on Networking (IFIP '12)*, Springer-Verlag, 2012, pp. 318–330.

[140] D. Halperin, et al., "Demystifying 802.11N Power Consumption". In *Proceedings of the 2010 International Conference on Power Aware Computing and Systems (HotPower '10)*, USENIX Association, 2010, Article 1, 5 pages.

[141] I. Pefkianakis, C. Li, S. Lu, "What is wrong/right with IEEE 802.11n Spatial Multiplexing Power Save feature?". In *Proceedings of the 19th IEEE International Conference on Network Protocols (ICNP '11)*, IEEE, 2011, pp. 186–195.

[142] N. Warty, R. K. Sheshadri, W. Zheng, D. Koutsonikolas, "A First Look at 802.11n Power Consumption in Smartphones". In *Proceedings of the 1st ACM MobiCom Workshop on Practical Issues and Applications in Next Generation Wireless Networks (PINGEN '12)*, ACM, 2012, pp. 27–32.

[143] A. Garcia-Saavedra, et al., "Energy Consumption Anatomy of 802.11 Devices and Its Implication on Modeling and Design". In *Proceedings of the 8th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT '12)*, ACM, 2012, pp. 169–180.

[144] R. Friedman, A. Kogan, Y. Krivolapov, "On Power and Throughput Tradeoffs of WiFi and Bluetooth in Smartphones". In *IEEE Transactions on Mobile Computing*, IEEE, 2013, Vol. 12, No. 7, pp. 1363–1376.

[145] Y. Zeng, P. H. Pathak, P. Mohapatra, "A first look at 802.11ac in action: Energy efficiency and interference characterization". In

*Proceedings of the IFIP Networking Conference (IFIP '14)*, IEEE, 2014, pp. 1–9.

[146] N. R. Potlapally, S. Ravi, A. Raghunathan, N. K. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols". In *IEEE Transactions on Mobile Computing*, IEEE, 2006, Vol. 5, No. 2, pp. 128–143.

[147] R. Chandramouli, S. Bapatla, K. P. Subbalakshmi, R. N. Uma, "Battery Power-Aware Encryption". In *ACM Transactions on Information and System Security*, ACM, 2006, Vol. 9, No. 2, pp. 162–180.

[148] Z. Guo, W. Jiang, N. Sang, Y. Ma, "Energy Measurement and Analysis of Security Algorithms for Embedded Systems". In *Proceedings of the IEEE/ACM International Conference on Green Computing and Communications (GreenCom '11)*, IEEE, 2011, pp. 194–199.

[149] P. Miranda, M. Siekkinen, H. Waris, "TLS and energy consumption on a mobile device: A measurement study". In *Proceedings of the IEEE Symposium on Computers and Communications (ISCC '11)*, IEEE, 2011, pp. 983–989.

[150] J. Flinn, M. Satyanarayanan, "PowerScope: A Tool for Profiling the Energy Usage of Mobile Applications". In *Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications (WMCSA '99)*, IEEE, 1999.

[151] F. Bellosa, "The Benefits of Event: Driven Energy Accounting in Power-sensitive Systems". In *Proceedings of the 9th Workshop on ACM SIGOPS European Workshop: Beyond the PC: New Challenges for the Operating System*, ACM, 2000, pp. 37–42.

[152] T. Li, L. K. John, "Run-time Modeling and Estimation of Operating System Power Consumption". In *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS '03)*, ACM, 2003, pp. 160–171.

[153] L. Zhang, et al., "Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones". In *Proceedings of the 8th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS '10)*, ACM, 2010, pp. 105–114.

[154] A. Pathak, et al., “Fine-grained Power Modeling for Smartphones Using System Call Tracing”. In *Proceedings of the Sixth Conference on Computer Systems (EuroSys ’11)*, ACM, 2011, pp. 153–168.

[155] M. Dong, L. Zhong, “Self-constructive High-rate System Energy Modeling for Battery-powered Mobile Systems”. In *Proceedings of the 9th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys ’11)*, ACM, 2011, pp. 335–348.

[156] W. Jung, et al., “DevScope: A Nonintrusive and Online Power Analysis Tool for Smartphone Hardware Components”. In *Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS ’12)*, ACM, 2012, pp. 353–362.

[157] F. Xu, Y. Liu, Q. Li, Y. Zhang, “V-edge: Fast Self-constructive Power Modeling of Smartphones Based on Battery Voltage Dynamics”. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation (NSDI ’13)*, USENIX Association, 2013, pp. 43–56.

[158] N. Vallina-Rodriguez, J. Crowcroft, “Energy Management Techniques in Modern Mobile Handsets”. In *IEEE Communications Surveys and Tutorials*, IEEE, 2013, Vol. 15, No. 1, pp. 179–198.

[159] A. Roy, et al., “Energy Management in Mobile Devices with the Cinder Operating System”. In *Proceedings of the Sixth Conference on Computer Systems (EuroSys ’11)*, ACM, 2011, pp. 139–152.

[160] N. Zeldovich, S. Boyd-Wickizer, E. Kohler, D. Mazières, “Making Information Flow Explicit in HiStar”. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI ’06)*, USENIX Association, 2006, pp. 263–278.

[161] N. Vallina-Rodriguez, J. Crowcroft, “ErdOS: Achieving Energy Savings in Mobile OS”. In *Proceedings of the Sixth ACM International Workshop on Mobility in the Evolving Internet Architecture (MobiArch ’11)*, ACM, 2011, pp. 37–42.

[162] H. Zeng, C. S. Ellis, A. R. Lebeck, A. Vahdat, “ECOSystem: Managing Energy As a First Class Operating System Resource”. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS X)*, ACM, 2002, pp. 123–132.

[163] J. Flinn, M. Satyanarayanan, “Managing Battery Lifetime with Energy-aware Adaptation”. In *ACM Transactions on Computer Systems*, ACM, 2004, Vol. 22, No. 2, pp. 137–179.

[164] D. Chu, A. Kansal, J. Liu, and F. Zhao, “Mobile Apps: It’s Time to Move Up to CondOS”. In *Proceedings of the 13th USENIX conference on Hot topics in operating systems (HotOS ’13)*, USENIX Association, 2011, 5 pages.

[165] I. Kelényi, J. K. Nurminen, “Bursty Content Sharing Mechanism for Energy-limited Mobile Devices”. In *Proceedings of the 4th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks (PM2HW2N ’09)*, ACM, 2009, pp. 216–223.

[166] M. Ra, et al., “Energy-delay Tradeoffs in Smartphone Applications”. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys ’10)*, ACM, 2010, pp. 255–270.

[167] C. Yoon, et al., “AppScope: Application Energy Metering Framework for Android Smartphones Using Kernel Activity Monitoring”. In *Proceedings of the 2012 USENIX Conference on Annual Technical Conference (USENIX ATC ’12)*, USENIX Association, 2012, pp. 387–400.

[168] L. Ravindranath, S. Agarwal, J. Padhye, C. Riederer, “Procrastinator: Pacing Mobile Apps’ Usage of the Network”. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys ’14)*, ACM, 2014, pp. 232–244.

[169] S. Mohapatra, N. Venkatasubramanian, “PARM : power aware reconfigurable middleware”. In *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS ’03)*, IEEE, 2003, pp. 312–319.

[170] M. Anand, E. B. Nightingale, J. Flinn, “Ghosts in the Machine: Interfaces for Better Power Management”. In *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services (MobiSys ’04)*, ACM, 2004, pp. 23–35.

[171] H. S. Ashwini, A. Thawani, Y. N. Srikant, “Middleware for Efficient Power Management in Mobile Devices”. In *Proceedings of the 3rd International Conference on Mobile Technology, Applications and Systems (Mobility ’06)*, ACM, 2006, Article 49.

[172] A. P. Miettinen, J. K. Nurminen, "Energy Efficiency of Mobile Clients in Cloud Computing". In *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud '10)*, USENIX Association, 2010.

[173] K. Kumar, Y. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?". In *IEEE Computer*, IEEE, 2010, Vol. 43, No. 4, pp. 51–56.

[174] Y. Xiao, P. Hui, P. Savolainen, 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 (MCS '11)*, ACM, 2011, pp. 13–18.

[175] A. Saarinen, et al., "Can Offloading Save Energy for Popular Apps?". In *Proceedings of the Seventh ACM International Workshop on Mobility in the Evolving Internet Architecture (MobiArch '12)*, ACM, 2012, pp. 3–10.

[176] D. Bertozzi, A. Raghunathan, L. Benini, and S. Ravi, "Transport Protocol Optimization for Energy Efficient Wireless Embedded Systems". In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '03)*, IEEE, 2003, pp. 706–711.

[177] C. Pluntke, L. Eggert, N. Kiukkonen, "Saving Mobile Device Energy with Multipath TCP". In *Proceedings of the Sixth ACM International Workshop on Mobility in the Evolving Internet Architecture (MobiArch '11)*, ACM, 2011, pp. 1–6.

[178] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses". Internet RFCs, ISSN 2070-1721, RFC 6824, 2013.

[179] E. Shih, P. Bahl, and M. J. Sinclair, "Wake on Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices". In *Proceedings of the 8th ACM International Conference on Mobile Computing and Networking (MobiCom '02)*, ACM, 2002, pp. 160–171.

[180] A. Rahmati, L. Zhong, "Context-for-Wireless: Context-Sensitive Energy-Efficient Wireless Data Transfer". In *Proceedings of the 5th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '07)*, ACM, 2007, pp. 165–178.

[181] C. Tsao, R. Sivakumar, "On Effectively Exploiting Multiple Wireless Interfaces in Mobile Hosts". In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT '09)*, ACM, 2009, pp. 337–348.

[182] T. Pering, Y. Agarwal, R. Gupta, R. Want, "CoolSpots: Reducing the Power Consumption of Wireless Mobile Devices with Multiple Radio Interfaces". In *Proceedings of the 4th International Conference on Mobile Systems, Applications, and Services (MobiSys '06)*, ACM, 2006, pp. 220–232.

[183] R. Krashinsky, H. Balakrishnan, "Minimizing Energy for Wireless Web Access with Bounded Slowdown". In *Wireless Networks*, Springer-Verlag, 2005, Vol. 11, No. 1-2, pp. 135–148.

[184] S. Nedevschi, et al., "Reducing Network Energy Consumption via Sleeping and Rate-adaptation". In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI '08)*, USENIX Association, 2008, pp. 323–336.

[185] Y. Agarwal, et al., "Wireless Wakeups Revisited: Energy Management for Voip over Wi-fi Smartphones". In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services (MobiSys '07)*, ACM, 2007, pp. 179–191.

[186] S. Mohapatra, et al., "A cross-layer approach for power-performance optimization in distributed mobile systems". In *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS '05)*, IEEE, 2005.

[187] S. Mohapatra, N. Dutt, A. Nicolau, N. Venkatasubramanian, "DYNAMO: A Cross-Layer Framework for End-to-End QoS and Energy Optimization in Mobile Handheld Devices". In *IEEE Journal on Selected Areas in Communications*, IEEE, 2007, Vol. 25, No. 4, pp. 722–737.

[188] F. Qian, et al., "Profiling Resource Usage for Mobile Applications: A Cross-layer Approach". In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services (MobiSys '11)*, ACM, 2011, pp. 321–334.

[189] B. D. Higgins, et al., "Informed Mobile Prefetching". In *Proceedings of the 10th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '12).* ACM, 2012, pp. 155–168.

[190] H. Liu, Y. Zhang, Y. Zhou, “TailTheft: Leveraging the Wasted Time for Saving Energy in Cellular Communications”. In *Proceedings of the Sixth ACM International Workshop on Mobility in the Evolving Internet Architecture (MobiArch ’11)*, ACM, 2011, pp. 31–36.

[191] A. Schulman, et al., “Bartendr: A Practical Approach to Energy-aware Cellular Data Scheduling”. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking (MobiCom ’2010)*, ACM, 2010, pp. 85–96.

[192] A. Chakraborty, V. Navda, V. N. Padmanabhan, R. Ramjee, “Coordinating Cellular Background Transfers Using Loadsense”. In *Proceedings of the 19th Annual International Conference on Mobile Computing and Networking (MobiCom ’13)*, ACM, 2013, pp. 63–74.

[193] S. Deng, H. Balakrishnan, “Traffic-aware Techniques to Reduce 3G/LTE Wireless Energy Consumption”. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies (CoNEXT ’12)*, ACM, 2012, pp. 181–192.

[194] R. K. Athivarapu, et al., “RadioJockey: Mining Program Execution to Optimize Cellular Radio Usage”. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (MobiCom ’12)*, ACM, 2012, pp. 101–112.

[195] L. Zhou, et al., “Performance Analysis of Power Saving Mechanism with Adjustable DRX Cycles in 3GPP LTE”. In *Proceedings of IEEE 68th Vehicular Technology Conference (VTC ’08)*, IEEE, 2008, pp. 1–5.

[196] A. T. Koc, S. C. Jha, R. Vannithamby, M. Torlak, “Device Power Saving and Latency Optimization in LTE-A Networks Through DRX Configuration”. In *IEEE Transactions on Wireless Communications*, IEEE, 2014, Vol. 13, No. 5, pp. 2614–2625.

[197] J. Liu, L. Zhong, “Micro Power Management of Active 802.11 Interfaces”. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services (MobiSys ’08)*, ACM, 2008, pp. 146–159.

[198] Z. Zeng, Y. Gao, P. R. Kumar, “SOFA: A Sleep-Optimal Fair-Attention Scheduler for the Power-Saving Mode of WLANs”. In *Proceedings of the 31st International Conference on Distributed Computing Systems (ICDCS ’11)*, IEEE, 2011, pp. 87–98.

[199] J. Manweiler, R. Roy Choudhury, "Avoiding the Rush Hours: WiFi Energy Management via Traffic Isolation". In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services (MobiSys '11)*, ACM, 2011, pp. 253–266.

[200] M. O. Khan, et al., "Model-driven Energy-aware Rate Adaptation". In *Proceedings of the Fourteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '13)*, ACM, 2013, pp. 217–226.

[201] F. R. Dogar, P. Steenkiste, K. Papagiannaki, "Catnap: Exploiting High Bandwidth Wireless Interfaces to Save Energy for Mobile Devices". In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, ACM, 2010, pp. 107–122.

[202] K. Jang, S. Hao, A. Sheth, R. Govindan, "Snooze: Energy Management in 802.11n WLANs". In *Proceedings of the Seventh Conference on Emerging Networking EXperiments and Technologies (CoNEXT '11)*, ACM, 2011, Article 12, 12 pages.

[203] E. Rozner, V. Navda, R. Ramjee, S. Rayanchu, "NAPman: Network-assisted Power Management for Wifi Devices". In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, ACM, 2010, pp. 91–106.

[204] N. Ding, et al., "Realizing the full potential of PSM using proxying". In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM '12)*, IEEE, 2012, pp. 2821–2825.

[205] A. Pathak, Y. C. Hu, M. Zhang, "Where is the Energy Spent Inside My App?: Fine Grained Energy Accounting on Smartphones with Eprof". In *Proceedings of the 7th ACM European Conference on Computer Systems (EuroSys '12)*, ACM, 2012, pp. 29–42.

[206] X. Ma, et al., "eDoctor: Automatically Diagnosing Abnormal Battery Drain Issues on Smartphones". In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation (NSDI '13)*, USENIX Association, 2013, pp. 57–70.

[207] N. Vallina-Rodriguez, et al., "RILAnalyzer: A Comprehensive 3G Monitor on Your Phone". In *Proceedings of the 2013 Conference on Internet Measurement Conference (IMC '13)*, ACM, 2013, pp. 257–264.

[208] A. J. Oliner, A. P. Iyer, I. Stoica, E. Lagerspetz, S. Tarkoma, “Carat: Collaborative Energy Diagnosis for Mobile Devices”. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys ’13)*, ACM, 2013, Article 10, 14 pages.

[209] E. Lagerspetz, “Collaborative Mobile Energy Awareness”. Department of Computer Science, University of Helsinki, Doctoral Thesis, Series of Publications A-2014-5, 2014.

[210] J. Paek, J. Kim, R. Govindan, “Energy-efficient Rate-adaptive GPS-based Positioning for Smartphones”. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys ’10)*, ACM, 2010, pp. 299–314.

[211] M. B. Kjaergaard, S. Bhattacharya, H. Blunck, P. Nurmi, “Energy-efficient Trajectory Tracking for Mobile Devices”. In *Proceedings of the 9th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys ’11)*, ACM, 2011, pp. 307–320.

[212] L. Song, et al., “Predictability of WLAN Mobility and Its Effects on Bandwidth Provisioning”. In *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM ’06)*, IEEE, 2006, pp. 1–13.

[213] J. Yoon, B. D. Noble, M. Liu, M. Kim, “Building Realistic Mobility Models from Coarse-grained Traces”. In *Proceedings of the 4th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys ’06)*, ACM, 2006, pp. 177–190.

[214] P. N. Pathirana, A. V. Savkin, S. Jha, “Mobility Modelling and Trajectory Prediction for Cellular Networks with Mobile Base Stations”. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc ’03)*, ACM, 2003, pp. 213–221.

[215] M. Kim, D. Kotz, S. Kim, “Extracting a Mobility Model from Real User Traces”. In *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM ’06)*, IEEE, 2006, pp. 1–13.

[216] M. C. González, C. A. Hidalgo, A. Barabási, “Understanding Individual Human Mobility Patterns”. In *Nature*, Nature Publishing Group, 2008, Vol. 453, pp. 779–782.

[217] M. Mathis, J. Semke, J. Mahdavi, T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm". In *ACM SIGCOMM Computer Communication Review*, ACM, 1997, Vol. 27, No. 3, pp. 67–82.

[218] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, "Modeling TCP Throughput: A Simple Model and Its Empirical Validation". In *Proceedings of the ACM SIGCOMM 1998 Conference (SIGCOMM '98)*, ACM, 1998, pp. 303–314.

[219] R. Enns, M. Bjorklund, J. Schoenwaelder, A. Bierman, "Network Configuration Protocol (NETCONF)". Internet RFCs, ISSN 2070-1721, RFC 6241, 2011.

[220] J. Case, M. Fedor, M. Schoffstall, J. Davin, "A Simple Network Management Protocol (SNMP)". Internet RFCs, ISSN 2070-1721, RFC 1157, 1990.

[221] N. McKeown, et al., "OpenFlow: Enabling Innovation in Campus Networks". In *ACM SIGCOMM Computer Communication Review*, ACM, 2008, Vol. 38, No. 2, pp. 69–74.

[222] S. Jain, et al., "B4: Experience with a Globally-deployed Software Defined Wan". In *Proceedings of the ACM SIGCOMM 2013 Conference (SIGCOMM '13)*, ACM, 2013, pp. 3–14.

[223] T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2". Internet RFCs, ISSN 2070-1721, RFC 5246, 2008.

[224] E. Rescorla, N. Modadugu, "Datagram Transport Layer Security Version 1.2". Internet RFCs, ISSN 2070-1721, RFC 6347, 2012.

[225] N. Gude, et al., "NOX: Towards an Operating System for Networks". In *SIGCOMM Computer Communication Review*, ACM, 2008, Vol. 38, No. 3, pp. 105–110.

[226] A. Tootoonchian, Y. Ganjali, "HyperFlow: A Distributed Control Plane for OpenFlow". In *Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking (INM/WREN '10)*, USENIX Association, 2010.

[227] S. Hassas Yeganeh, Y. Ganjali, "Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications". In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN '12)*, ACM, 2012, pp. 19–24.

[228] T. Koponen, et al., "Onix: A Distributed Control Platform for Large-scale Production Networks". In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI '10)*, USENIX Association, 2010, pp. 1–6.

[229] A. Dixit, et al., "Towards an Elastic Distributed SDN Controller". In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN '13)*, ACM, 2013, pp. 7–12.

[230] L. E. Li, Z. M. Mao, J. Rexford, "Toward Software-Defined Cellular Networks". In *Proceedings of European Workshop on Software Defined Networking (EWSDN '12)*, IEEE, 2012, pp. 7–12.

[231] K. Yap, et al., "Blueprint for Introducing Innovation into Wireless Mobile Networks". In *Proceedings of the Second ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA '10)*, ACM, 2010, pp. 25–32.

[232] A. Gudipati, D. Perry, L. E. Li, S. Katti, "SoftRAN: Software Defined Radio Access Network". In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN '13)*, ACM, 2013, pp. 25–30.

[233] M. Bansal, J. Mehlman, S. Katti, P. Levis, "OpenRadio: A Programmable Wireless Dataplane". In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN '12)*, ACM, 2012, pp. 109–114.

[234] X. Jin, L. E. Li, L. Vanbever, J. Rexford, "SoftCell: Scalable and Flexible Cellular Core Network Architecture". In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT '13)*, ACM, 2013, pp. 163–174.

[235] L. Suresh, et al., "Towards Programmable Enterprise WLANS with Odin". In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN '12)*, ACM, 2012, pp. 115–120.

[236] V. Sivaraman, et al., "Virtualizing the Access Network via Open APIs". In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT '13)*, ACM, 2013, pp. 31–42.

[237] S. Shirali-Shahreza, Y. Ganjali, "Efficient Implementation of Security Applications in OpenFlow Controller with FleXam". In *Proceedings*

*of the 21st Annual Symposium on High-Performance Interconnects (HOTI '13)*, IEEE, 2013, pp. 49–54.

[238] N. Dukkipati, et al., "An Argument for Increasing TCP's Initial Congestion Window". In *SIGCOMM Computer Communication Review*, ACM, 2010, Vol. 40, No. 3, pp. 26–33.

[239] J. Chu, N. Dukkipati, Y. Cheng, M. Mathis, "Increasing TCP's Initial Window". Internet RFCs, ISSN 2070-1721, RFC 6928, 2013.

[240] H. Jiang, et al., "Understanding Bufferbloat in Cellular Networks". In *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design (CellNet '12)*, ACM, 2012, pp. 1–6.

[241] J. Gettys, K. Nichols, "Bufferbloat: Dark Buffers in the Internet". In *ACM Queue*, ACM, 2011, Vol. 9, No. 11.

[242] S. Floyd, V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance". In *IEEE/ACM Transactions on Networking*, IEEE, 1993, Vol. 1, No. 4, pp. 397–413.

[243] Technology Vision 2020 White Paper. Nokia Networks.

[244] D. Thaler and R. Draves, A. Matsumoto, T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)" Internet RFCs, ISSN 2070-1721, RFC 6724, 2012.

[245] D. Thaler and B. Aboba, "What Makes for a Successful Protocol?" Internet RFCs, ISSN 2070-1721, RFC 5218, 2008.

[246] Y. Ding, T. Savolainen, J. Korhonen, M. Kojo, "Speeding up IPv6 Transition: Discovering NAT64 and Learning Prefix for IPv6 Address Synthesis". In *Proceedings of IEEE International Conference on Communications (ICC '12)*, IEEE, 2012, pp. 6862–6868.

[247] A. Y. Ding, J. Korhonen, T. Savolainen, Y. Liu, M. Kojo, S. Tarkoma, H. Schulzrinne, "Reflections on Middlebox Detection Mechanisms in IPv6 Transition". In *IAB Workshop on Stack Evolution in a Middlebox Internet (SEMI '15)*, Internet Architecture Board, 2015, 11 pages.

[248] W. Dec, T. Mrugalski, T. Sun, B. Sarikaya, "DHCPv6 Route Option". Internet-Draft "draft-dec-dhcpv6-route-option-05.txt", work-in-progress, IETF Secretariat, September 2010.

[249] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)" Internet RFCs, ISSN 2070-1721, RFC 3315, 2003.

[250] T. Narten, E. Nordmark, W. Simpson, H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)" Internet RFCs, ISSN 2070-1721, RFC 4861, 2007.

[251] C. Perkins, D. Johnson, J. Arkko, "Mobility Support in IPv6" Internet RFCs, ISSN 2070-1721, RFC 6275, 2011.

[252] A. Conta, S. Deering, M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification" Internet RFCs, ISSN 2070-1721, RFC 4443, 2006.

[253] T. Savolainen, J. Kato, and T. Lemon, "Improved Recursive DNS Server Selection for Multi-Interfaced Nodes" Internet RFCs, ISSN 2070-1721, RFC 6731, 2012.

[254] C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)" Internet RFCs, ISSN 2070-1721, RFC 7296, 2014.

[255] IEEE Standards Association. "802.11i-2004 - IEEE Standard for information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 6: Medium Access Control (MAC) Security Enhancements". IEEE Std 802.11i, 2004.

[256] IEEE Standards Association. "802.11-1997 - IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". IEEE Std 802.11, 1997.

[257] D. Hadaller, S. Keshav, T. Brecht, and S. Agarwal, "Vehicular Opportunistic Communication Under the Microscope". In *Proceedings of the 5th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '04)*, ACM, 2007, pp. 206–219.

[258] V. Navda, A. P. Subramanian, K. Dhanasekaran, A. Timm-Giel, S. Das, "MobiSteer: Using Steerable Beam Directional Antenna for Vehicular Network Access". In *Proceedings of the 5th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '04)*, ACM, 2007, pp. 192–205.

[259] E. C. Efstathiou, P. A. Frangoudis, C. G. Polyzos, "Controlled Wi-Fi Sharing in Cities: A Decentralized Approach Relying on Indirect Reciprocity". In *IEEE Transactions on Mobile Computing*, IEEE, 2010, Vol. 9, No. 8, pp. 1147–1160.

[260] K. Lin, A. Kansal, D. Lymberopoulos, F. Zhao, "Energy-Accuracy Trade-off for Continuous Mobile Device Location". In *Proceedings of the 8th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, ACM, 2010, pp. 285–298.

[261] H. Liu, et al., "Push the Limit of WiFi based Localization for Smartphones". In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (MobiCom '12)*, ACM, 2012, pp. 305–316.

[262] V. Valancius, et al., "Greening the Internet with Nano Data Centers". In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT '09)*, ACM, 2009, pp. 37–48.

[263] J. Yun, K. G. Shin, "CTRL: A Self-organizing Femtocell Management Architecture for Co-channel Deployment". In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking (MobiCom '2010)*, ACM, 2010, pp. 61–72.

[264] J. Huang, et al., "Anatomizing Application Performance Differences on Smartphones". In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, ACM, 2010, pp. 165–178.

[265] N. Sastry, J. Crowcroft, and K. Sollins. "Architecting Citywide Ubiquitous Wi-Fi Access". In *Proceedings of Workshop on Hot Topics in Networks (HotNets '07)*, 2007, pp. 1–7.

[266] D. Leroy, G. Detal, J. Cathalo, M. Manulis, F. Koeune, O. Bonaventure. "SWISH: Secure WiFi Sharing". *Computer Networks*, Elsevier, 2011, Vol. 55, No. 7, pp. 1614–1630.

[267] R. Cáceres, V. N. Padmanabhan, “Fast and Scalable Handoffs for Wireless Internetworks”. In *Proceedings of the 2nd Annual International Conference on Mobile Computing and Networking (MobiCom ’96)*, ACM, 1996, pp. 56–66.

[268] J. Mysore, V. Bharghavan, “A New Multicasting-based Architecture for Internet Host Mobility”. In *Proceedings of the 3rd Annual International Conference on Mobile Computing and Networking (MobiCom ’97)*, ACM, 1997, pp. 161–172.

[269] A. C. Snoeren, H. Balakrishnan, “An End-to-end Approach to Host Mobility”. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom ’00)*, ACM, 2000, pp. 155–166.

[270] S. Zhuang, et al., “Host Mobility Using an Internet Indirection Infrastructure”. In *Proceedings of the 1st International Conference on Mobile Systems, Applications, and Services (MobiSys ’03)*, ACM, 2003, pp. 129–144.

[271] I. Stoica, et al., “Internet Indirection Infrastructure”. In *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM ’02)*, ACM, 2002, pp. 73–86.

[272] M. Buddhikot, et al., “Integration of 802.11 and Third-Generation Wireless Data Networks”. In *Proceedings of the 22nd IEEE International Conference on Computer Communications (INFOCOM ’03)*, IEEE, 2003, pp. 503–512.

[273] T. Armstrong, O. Trescases, C. Amza, E. de Lara, “Efficient and Transparent Dynamic Content Updates for Mobile Clients”. In *Proceedings of the 4th International Conference on Mobile Systems, Applications, and Services (MobiSys ’06)*, ACM, 2006, pp. 56–68.

[274] B. D. Higgins, et al., “Intentional Networking: Opportunistic Exploitation of Mobile Network Diversity”. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking (MobiCom ’2010)*, ACM, 2010, pp. 73–84.

[275] Y. Zhang, “User Mobility from the View of Cellular Data Networks”. In *Proceedings of the 33rd IEEE International Conference on Computer Communications (INFOCOM ’14)*, IEEE, 2014, pp. 1348–1356.

[276] H. Haverinen, J. Salowey, "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)". Internet RFCs, ISSN 2070-1721, RFC 4186, 2006.

[277] IEEE Standards Association. "802.11u-2011 - IEEE Standard for Information Technology-Telecommunications and information exchange between systems-Local and Metropolitan networks-specific requirements-Part II: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 9: Interworking with External Networks". IEEE, 2011.

[278] S. Taylor, A. Young, A. Noronha, "What Do Consumers Want from Wi-Fi? Insights from Cisco IBSG Consumer Research". Cisco Internet Business Solutions Group (IBSG) White Paper, 2012.

[279] C. Hetting, "Seamless Wi-Fi Offload: A Business Opportunity Today". APTILO NETWORKS WHITE PAPER, 2013.

[280] Wireless Broadband Alliance, "Global Developments in Public Wi-Fi". WBA Industry Report, 2011.

[281] E. Kohler, et al., "The Click Modular Router". In *Transactions on Computer Systems*, ACM, 2000, Vol. 18, No. 3.

[282] S. Hemminki, P. Nurmi, S. Tarkoma, "Accelerometer-based Transportation Mode Detection on Smartphones". In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys '13)*, ACM, 2013, article no. 13.

[283] R. Mahindra, H. Viswanathan, K. Sundaresan, M. Y. Arslan, S. Rangarjan, "A Practical Traffic Management System for Integrated LTE-WiFi Networks". In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom '14)*, ACM, 2014, pp. 189–200.

[284] A. Patro, S. Banerjee, "Outsourcing Coordination and Management of Home Wireless Access Points through an Open API". In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM '15)*, IEEE, 2015.

[285] I. F. Akyildiz, P. Wang, S. Lin, "SoftAir: A software defined networking architecture for 5G wireless systems". In *Computer Networks*, Elsevier, 2015, Vol. 85, pp. 1–18.

[286] M. Amani, T. Mahmoodi, M. Tatipamula, H. Aghvami, "Programmable policies for data offloading in LTE network". In *Proceedings of the IEEE International Conference on Communications (ICC '14)*, IEEE, 2014, pp. 3154–3159.

[287] J. Cho, et al., "SMORE: Software-defined Networking Mobile Offloading Architecture". In *Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications, and Challenges*, ACM, 2014, pp. 21–26.