On the Performance of a Flow Aggregation Scheme for Seamless QoS and Utility Oriented Mobility Support in Wireless Mesh Networks

Salvatore Vanini*, Dario Gallucci*, Silvia Giordano*, Maciej Urbanski†, and Przemyslaw Walkowiak†
*Information Systems and Networking Institute
SUPSI, Manno, Switzerland
Email: [salvatore.vanini,dario.gallucci,silvia.giordano]@supsi.ch
†Institute of Control and Information Engineering
Poznan University of Technology, Poznan, Poland
Email: [maciej.urbanski,przemyslaw.walkowiak]@put.poznan.pl

***Abstract*—Wireless Mesh Networks are a viable solution for extending the coverage of networks without the need to expand their static infrastructure. This is achieved by employing wireless nodes that can share their network resources. Unfortunately, building wireless mesh networks is still far away from reality due to the lack of mechanisms to meet the Quality of Service (QoS) expectations of mobile users running heterogeneous applications. To address this issue, in this paper we describe a modification of the WiOptiMo framework, which is a solution originally designed for seamless handover management in the Internet. Specifically, QoS support is provided by aggregating application traffic flows with the same characteristics to limit overhead and by relaying compressed aggregated flows to the appropriate mobility provider. Evaluation on a real wireless mesh network testbed showed that this scheme is noteworthy in terms of link utilization, improved QoS and performance against Mobile IPv6, which is the standard for enabling mobility in IP networks. To show its adaptability, we present a scenario where WiOptiMo can be employed to support network mobility when the range of a network is extended by exploiting neighboring nodes. Beside the evidence of performance gain against Mobile IPv6, the other contribution of this paper is the proposal of an incentive mechanism to motivate users of a wireless mesh network to share their network resources with other nodes, which rewards them based on the amount of bytes saved thanks to the use of the WiOptiMo flow aggregation scheme.**

Keywords–Wireless Mesh Networks; Seamless Handover; QoS Mobility Support; Flow Aggregation; Flow Classification; Mobile IP; Network Utility Maximisation.

I. INTRODUCTION

This paper builds on the work [1] (presented at MOBILITY 2014), which dealt with the design, implementation and evaluation of a flow classification and aggregation scheme for managing multiple applications with different Quality of Service (QoS) requirements in Wireless Mesh Networks (WMNs).

Recent years have witnessed a significant reduction in the costs of mobile computing platforms (e.g., laptops and smartphones), especially the hardware used in WiFi devices and has led to a widespread use of WMNs. WMNs provide multiple services to people using their mobile devices via a combination of fixed and mobile nodes, interconnected via wireless links to form a multi-hop ad-hoc network. WMNs are a cost-effective solution to extend the range of wired infrastructure networks with the help of easy to deploy wireless nodes. For example, the backbone of a telecom service provider can be easily expanded utilizing mechanisms to manage resources of wireless nodes [2][3]. Existing mechanisms work only in scenarios where wireless connection stability can be ensured. For example, CARMNET [4][5] utilizes the WMN paradigm to enable nearby wireless devices communicate with each other and proposes a distributed resource management method that can be easily integrated with a telecom IP-based Multimedia Subsystem (IMS) software infrastructure. This method (implicitly) assumes that the underlying network connectivity is not affected by topological changes (e.g., gateway changes) caused by the mobility of network's nodes. During those changes, packets for a given application flow might be rejected because of the change of the IP address, or they might be lost due to out-dated routing information. As a consequence, the quality and performance of correspondent applications can significantly decrease. The main protocols that can be used for supporting network mobility in IP-based WMN architectures are Mobile IPv4 (MIPv4) [6] and Mobile IPv6 (MIPv6) [7]. Mobile IP focuses on keeping the IP identity of a mobile node. MIPv6 is an enhancement of MIPv4 in terms of performance, however it might have long delay (handover latency) and high packet loss rate because of signaling traffic overhead. For this reason, several extensions to the MIPv6 base protocol, such as Fast Handovers for MIPv6 [8] and Hierarchical MIPv6 [9], have been proposed to get a better performance. A completely different approach to network mobility support is the use of schemes adopted in pure ad-hoc networks, which focus on rerouting (i.e., finding an alternative path in a timely manner, so that a flow can be handed off to the new path upon link disruption). Unfortunately, these mechanisms performs poorly in WMNs. To overcome the limitations of these approaches, several works have proposed different schemes for providing QoS and seamless mobility support in WMNs. However, many of them are not designed to manage multimedia services with QoS requirements—e.g., Voice over IP (VoIP) or Video on Demand (VoD).

This work is organized as follows. In Section II, we describe the previous work on mobility management in WMNs. In Section III, we present an extension of our WiOptiMo [10] framework to provide generalized QoS mobility support in WMNs. In Sections IV and V, we describe our enhanced framework and flow aggregation scheme to provide the required QoS to different types of applications in a WMN scenario. Then, in Section VI, we evaluate the performance improvement with respect to its typical configuration for WMNs. In Section VII, we compare the performance of

WiOptiMo with the standard MIPv6 implementation. Then, in Section VIII, we briefly describe the CARMNET WMN architecture - which provides mechanisms for sharing network access and extending network coverage - and present a scenario where WiOptiMo is used to bridge the signal gap between different access points and to provide handover functionality. Finally, in the context of a CARMNET WMN architecture, we propose an incentive mechanism for users to share their network access, which is based on the amount of bytes saved thanks to the use of the WiOptiMo aggregation scheme.

II. Related Work

The existing work on mobility management in WMNs focuses on providing network-layer mobility support. RFC 4886 [11] specifically addresses the issue of network mobility. The main protocol used for mobility management at the IP layer is MIPv6 [7]. However, MIPv6 has some well known drawbacks such as signaling traffic overhead. This results in long delay (handover latency) and high packet loss rate, thereby causing a QoS deterioration of real-time traffic. Furthermore, MIPv6 has some scalability problems that arise since it handles mobile node local mobility in the same way as global mobility. For these reasons, several extensions of MIPv6, such as Fast Handovers for MIPv6 [8] and Hierarchical MIPv6 [9], have been introduced to increase its performance. Fast Handovers for MIPv6 was proposed to reduce the handover latency by providing IP connectivity as soon as a mobile node attaches to a new subnet. To realize this, a mobile node performs a probe task to discover nearby access points. The main drawback of this process is that the mobile node cannot receive or send data during the probe phase. HMIPv6 was proposed to handle handover locally, thereby reducing unnecessary signaling overhead and latency within a domain, but suffering from same delay for global communication. To sum up, despite MIPv6 extensions, mobility management with QoS provision in WMNs remains a challenging task.

Interworking between 3GPP cellular network and WLAN is addressed by the Third Generation Partnership Program (3GPP), which developed an architecture to enable 3GPP cellular network subscribers to access WLAN service [12]. The interworking architecture provides fast deployment for global roaming and billing. This initiative is focused on specific standards and its standardization is currently ongoing [13].

The different solutions presented in literature focus on managing the address of a mobile node due to the handover process. In general, we can distinguish between intra-domain and inter-domain mobility. The first refers to handovers inside the same network domain, the second to handovers between different network domains. MobileNAT [14] addresses both intra- and inter-domain mobility. It allows a mobile node to keep a fixed IP address as it roams across the same or a different domain. MobileNAT requires a modification at the network layer stack of a mobile node and changes to the standard DHCP protocol, which introduces network latency. SyncScan [15] is a Layer-2 procedure for intra-domain handover in 802.11 infrastructure mode networks. It achieves good performance at the expense of a required global synchronization of beacon timings between clients and access points (AP). iMesh [16] provides low handover latency for Layer-3 intra-domain handovers between APs of a WMN. However, the handover latency depends on the number of nodes between the new and old AP. BASH [17] focuses on the design of a intra-domain Layer-2 seamless handover scheme for 802.11 WMNs, but the handover protocol requires modifications at every mobile client. Authors of [18] use tunneling, as well as the standard Mobile IPv6 solution [7] and most of the existing network-layer mobility management schemes based on Mobile IP, such as Mobile Party [19] and AODV-PRD [20]. Tunneling introduces extra delay for the encapsulation/decapsulation of packets and has intrinsically low flexibility. Finally, SMesh [21] provides a 802.11 mesh network architecture for both intra-domain and inter-domain handovers. For intra-domain handovers, SMesh generates high network overhead, which grows linearly with the number of mobile clients. In case of inter-domain handovers, network overhead generated by SMesh is proportional to the number of connections of a mobile client. The WiOptiMo framework provides mobility support by separately managing each application's flow, to meet the QoS expectations of all applications. In [10], we describe the architecture of WiOptiMo and in [22] we present how it is adapted to handle a WMN context. In the next sections, we show how its architecture has been modified to handle efficiently multiple application's flows with different QoS requirements and improve performance of standard mobility management mechanisms.

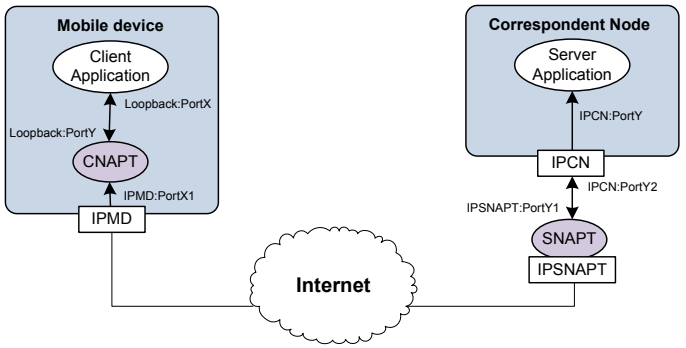

Figure 1. WiOptiMo's CNAPT and SNAPT IP decoupling.

III. The WiOptiMo framework

WiOptiMo handles IP network mobility and enables handovers initiated by a mobile device. It manages the mobility of every device with the help of two software modules: Client Network Address & Port Translator (CNAPT) and Server Network Address & Port Translator (SNAPT). Together, these two components provide decoupling between the IP address assigned to a mobile device and the IP address used to access a service on the Internet. CNAPT and SNAPT hide any change of the IP address when a mobile host moves between different access networks, inside the same domain or between different domains. In Figure 1, we describe the basic idea of the WiOptiMo framework. A mobile device with IP address IPMD has an active TCP session to a corresponding node with IP address IPCN. The TCP data packets are first relayed to the local CNAPT, which in turn relays them to the SNAPT. Upon receiving packets, the SNAPT (processes and) forwards them to the IPCN address. When the mobile device moves to a new network and gets a new IP address, the change in IP address does not affect the application layer because the application packets are sent to the the local CNAPT, which relays them to

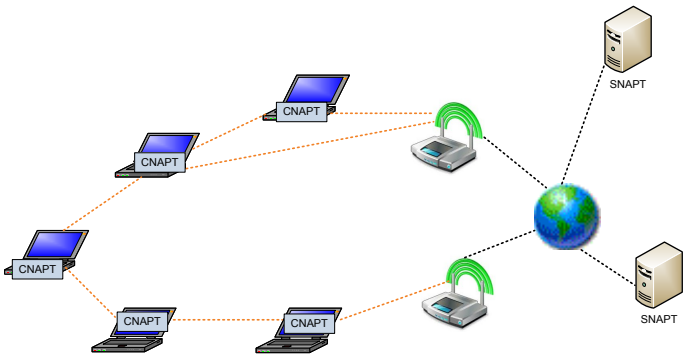

Figure 2. WiOptiMo configuration for a WMN.

the SNAPT with fixed IP address (IPSNAPT). This mechanism also allows a mobile node of a WMN to change gateway transparently (e.g., when node moves out of the reach of the initial gateway due to the mobility of the associated user), without suffering service disruption. To correctly manage the handover process, CNAPT and SNAPT exchange handshaking packets with each other using a control socket.

In a generalized setting, mobile devices have CNAPT installed on them, while an Internet server or any node in a network (as in the scenario previously described) have SNAPT installed on them.

A. WiOptiMo Architecture for a WMN

In [23], we present a general configuration of our WiOptiMo for a WMN. We exploit the flexibility of location where a SNAPT can be installed to address scalability issues that might arise in a WMN. In this scenario, multiple SNAPTs can be deployed on mesh routers or on Internet nodes to avoid network congestion in a single spot. This solution overcomes the scalability issues of MIPv6 because local mobility can be managed in a more efficient way. Every mobile wireless device has CNAPT installed on it to provide independent mobility support. We use a combination of network status monitoring and user configurable policy to enable every CNAPT to choose a suitable SNAPT that will relay its application flows. At start-up, each CNAPT connects to a fixed SNAPT specified in a configuration file. Then, it receives a list of other available SNAPTs from the currently connected SNAPT, and measures the delay towards them by means of passive and active monitoring of the control connection towards the SNAPTs, used for handshaking. CNAPTs also take into account the bandwidth used by applications in order to make a more wise SNAPT choice. The CNAPTs select a SNAPT to relay their data depending on the measured delay and estimated remaining throughput (based on the application's bandwidth requirements). This selection policy also helps in reducing the overload on any single SNAPT. However, there is still a limitation due to the architecture of Internet routing: it is not possible to change the SNAPT handling an application until its data connections end. Figure 2 shows WiOptiMo's architecture for a WMN. The SNAPTs can be managed by private administrators (otherwise called mobility service providers), who may require a fee for the use of their mobility service. This circumstance might foster the competition between mobility service providers, forcing them to increase the quality of provided service and benefit the entire WMN.

B. Implementation changes

We adapted WiOptiMo's implementation (both CNAPT and SNAPT) for low profile devices and to provide a fast handover procedure. Figure 3 shows the changes to the basic implementation of WiOptiMo. A TCP control socket still manages the communication between a CNAPT and a SNAPT. It provides network configuration parameters (e.g., the Maximum Transmission Unit - MTU - of the underlying network) and also transmits data packets in a fall-back mode when middle-boxes, such as firewalls and/or Network Address Translation (NAT), block UDP packets. Further, the control socket is used to authenticate the CNAPT and to exchange a session key for providing data authenticity and integrity during a handover. The CNAPT relays data packets to SNAPTs (and vice versa) using UDP sockets—this solution increases performance during handovers, because UDP does not need to retransmit lost packets nor does it perform any connection setup. When a SNAPT receives a UDP data packet, it validates it using HMAC [24] and tests it against replay attacks using a sequence number. During handovers (i.e., when the source IP address of data packets changes), the SNAPT updates the return IP address for the flow and transmits a keep-alive request to the CNAPT, which will reset the control connection or hasten the detection of a timeout. This event will then trigger the re-establishment of the control socket connection to the SNAPT.

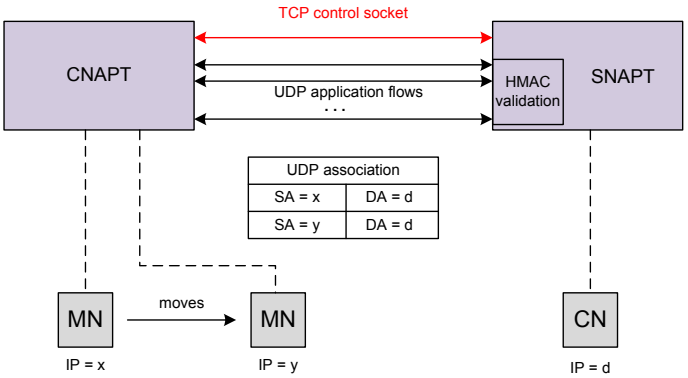

Figure 3. WiOptiMo adaptation for a WMN.

IV. QoS SUPPORT IN WIOPTIMO

In the next sections, we summarize the underlying mechanism and the main experimental results in terms of performance of the flow classification and aggregation scheme in our WiOptiMo framework that we presented in [1].

A. Flow classification

Since WiOptiMo relays each outgoing data flow from a client to a server application (through the link between CNAPT and SNAPT), every flow from a mobile device to its intended destination can be managed separately, according to its characteristics. We exploit this capability to relay data flows to different SNAPTs based on their delay and throughput needs, this way meeting the QoS requirements of applications. In this regard, we identified four different flow classes according to the minimum throughput and maximum delay requirements of applications: *High Throughput and High Delay* (HT & HD),

High Throughput and Low Delay (HT & LD), *Low Throughput and High Delay* (LT & HD), *Low Throughput and Low Delay* (LT & LD). Table I presents performance requirements of the most popular applications, along with their correspondent classification. In terms of throughput, the minimum threshold for classifying HT flow classes is 64kbit/s. We set the maximum delay for LD classes to 1s. As previously stated, during the normal workflow, a CNAPT periodically measures delay (one-trip time) and throughput (amount of received data over a time period) towards the different SNAPTs. Then, for each application flow, it detects the class type on the basis of process name, protocol and port number. Every class has an assigned delay and throughput requirements, and data flows get relayed to a SNAPT that meets their delay and throughput requirements.

TABLE I. Applications requirements based on throughput and delay, and their classification.

Application	Class	Min throughput/Max delay
Skype / Video and Voice	HT & LD	128kbit/s / 200ms
Skype only Voice	HT & LD	30kbit/s / 500ms
SSH Client	LT & LD	10kbit/s / 200ms
Web Browser	HT & HD	- / 5s
FTP Client	HT & HD	- / 5s
Google Hangout Video	HT & LD	256kbit/s / 200ms
Google Hangout Chat	LT & LD	10kbit/s / 3s
Remote Desktop Client	LT & LD	- / 200-500ms
Team Viewer	HT & LD	- / 200-500ms
Applets / Widgets	LT & HD	- / 10-30s
Default TCP	LT & HD	- / -
Default UDP	HT & LD	- / -

While our solution for flow classification is conceptually similar to DiffServ [25], it does not have its drawbacks. First, flow classification is performed dynamically per SNAPT, so that new flows are allocated depending on the current network performance statistics (e.g., the increase of the delay with the increase of the load). Second, our framework might refuse to serve a flow if its QoS requirements cannot be met, hence avoiding to disrupt the traffic already allocated. Moreover, the routing layer, as explained in [23], knows which traffic is managed by WiOptiMo. In this way, a QoS-aware routing mechanism can be executed whenever needed. In particular, network statistics about each single flow are reported to the routing layer so that there is no loss of granularity in the traffic management.

V. Flow Aggregation Mechanism

The class based aggregation technique implemented in our WiOptiMo framework allows to enhance its performance, to efficiently handle applications flows with short frequent sessions (e.g., DNS requests), to optimize wireless link utilization and to increase fairness between competing flows (which is a major drawback when wireless links have high latency [26]). Classified flows that belong to the same class are treated as a single aggregate and transmitted to a SNAPT using the same UDP socket. Our objective is to maximize the utilization of the available link bandwidth and reduce network overhead, thereby increasing the achieved throughput without significantly impacting the latency requirements.

Figure 4 presents the details of our aggregation mechanism. We implemented four connection queues, one for each of the application classes defined in Section IV-A. The queues feed into a scheduler, which uses a connection strategy based on flows' priority: the scheduler sends classes with more stringent requirements in terms first of delay and then of bandwidth — this is implemented as a simple static priority queue, cycling through LT & LD , HT & LD , LT & HD and HT & HD queues at dynamic intervals, depending on the processed traffic. To reduce the amount of exchanged data, we enabled compression of the aggregated flows — packets are appended to the aggregated compound until their cumulative compressed size does not exceed the 70% of the underlying network's MTU. We chose this threshold to maximize the effectiveness of aggregation without having to resort to a slower algorithm.

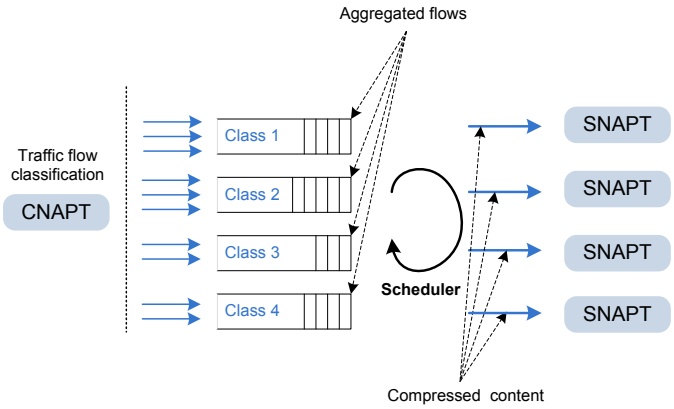

Figure 4. Software architecture of the aggregation scheme.

TABLE II. Different Parameters of the Experiment.

Application Class	Packet Size (Bytes)	Range of bit-rate (bit/s)	Range of Flows
HT & HD	1360	1M - 20M	1 - 5
HT & LD	576	128k - 2M	1 - 5
LT & HD	1360	15k - 1M	1 - 5
LT & LD	100	15k - 128k	1 - 5

HT - High Throughput LT - Low Throughput
HD - High Delay LD - Low Delay

VI. Performance of the Flow Aggregation Scheme

We experimentally assessed the performance and QoS support of WiOptiMo with flow aggregation.

A. Performance of WiOptiMo with flow aggregation

To evaluate the performance of our flow aggregation scheme, we conducted experiments in three different scenarios:

1) Baseline: without WiOptiMo.
2) WiOptiMo basic.
3) WiOptiMo with flow aggregation mechanism.

Measurements showed that the performance of the baseline and WiOptiMo basic configurations are comparable (the degradation on throughput and the additional end-to-end delay introduced by the WiOptiMo solution are negligible, as presented also in [10]). For this reason, we report only the results for the baseline and WiOptiMo with flow aggregation scenarios, and show that our flow aggregation scheme achieves a better link utilization and reduces the amount of bytes exchanged in the network.

Our experiment setup was composed by a WiOptiMo SNAPT server and a WiOptiMo CNAPT client (installed on a Dell Precision M4300 with LinkSys Dual-Band Wireless A+G PCI Card), connected through a Netgear WNDR3800 wireless router (with OpenWRT 12.09) To avoid interference with nearby 802.11 access points operating on the 2.4 GHz band, we enabled only 802.11a networking on our router. Both client and server operated on a Linux distribution (Ubuntu 12.04 with Linux kernel 3.11).

We used the *Iperf* [27] network testing tool to send a stream of UDP packets (at a specific bit-rate) to server and measured the number of bytes sent between client and server using the *dumpcap* utility [28]. Instead of using the default UDP packets generated by Iperf—all packets contain same data—we configured the Iperf utility to generate UDP packets containing random text stored in a file. We performed experiments under the four different classes described in Section IV-A. For each flow class, we fixed the size of data in every UDP packet transmitted by the Iperf utility. We repeated experiments 10 times, to get more reliable results. Table II shows the characteristics of every flow generated by Iperf to measure the performance of WiOptiMo (for each application class).

We measured the performance of WiOptiMo by varying the number of flows and bit-rate of each flow, and observing their impact on the percentage of bytes saved on the link, due to flow aggregation and compression. The last is calculated by subtracting pre-aggregation (and compression) bytes and post-aggregation (and compression) bytes, and dividing this difference by the pre-aggregation (and compression) bytes. This metric measures the bytes saved in the packet transfer between the client and server with the flow aggregation configuration, compared to the baseline configuration. It captures the energy spent to transfer data to the server. Since WiOptiMo performs flow aggregation and compression, this metric will enable us to measure the amount of energy that could be saved without impacting the QoS of applications.

Figure 5 shows the percentage of bytes saved for applications with high throughput and high delay network requirements. We observe that for bit rates lower than 10Mbit/s, the percentage of bytes saved increases as the number of flows increases. Even for a single application flow, WiOptiMo

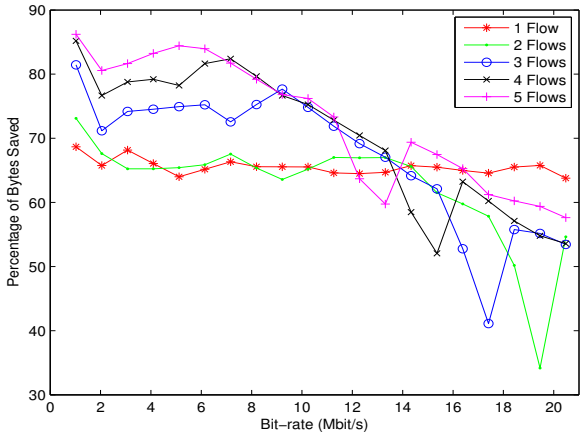

Figure 5. Percentage of bytes saved due to flow aggregation in HT & HD applications.

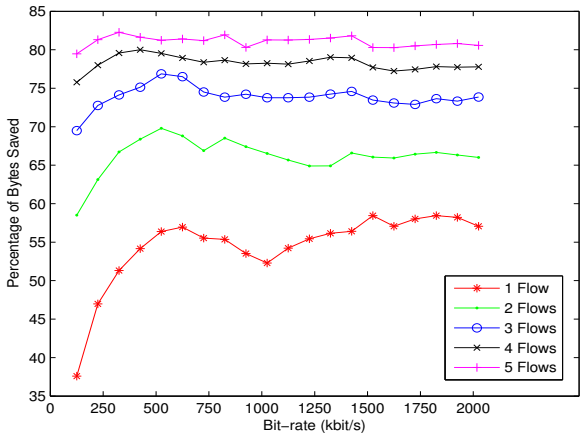

Figure 6. Percentage of bytes saved due to flow aggregation in HT & LD applications.

with flow classification and aggregation helps in reducing, on average, (down) to 60% the amount of data exchanged between client and server. For bit-rates higher than 10Mbit/s, the percentage of bytes saved is still high but its relationship with the number of flows is no longer linear. This behaviour is due to the saturation of the system's modules capacity (wireless card, aggregation and compression mechanisms).

In Figure 6, we observe that when applications have high throughput and low delay requirements, savings by WiOptiMo increase from 38% for single flow to a maximum of 82.5% for applications with 5 flows. For all flows, the percentage of bytes saved increases until the bit-rate reaches about 400kbit/s. For much higher rates we observe that the percentage of bytes saved remains constant.

For low throughput and high delay tolerant applications (see Figure 7), we observe that for low bit-rates (∼125kbit/s), the percentage of bytes saved is not significant because no additional savings could be achieved by compressing and aggregating data packets arriving at long intervals of time. For higher bit rates (that is after the size of the aggregated packets allows better compression), savings increase and then

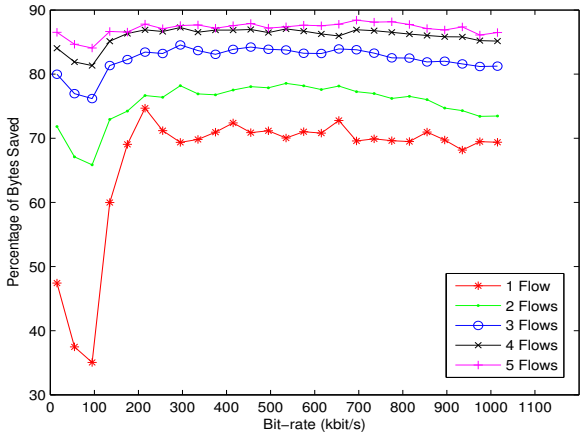

Figure 7. Percentage of bytes saved due to flow aggregation in LT & HD applications.

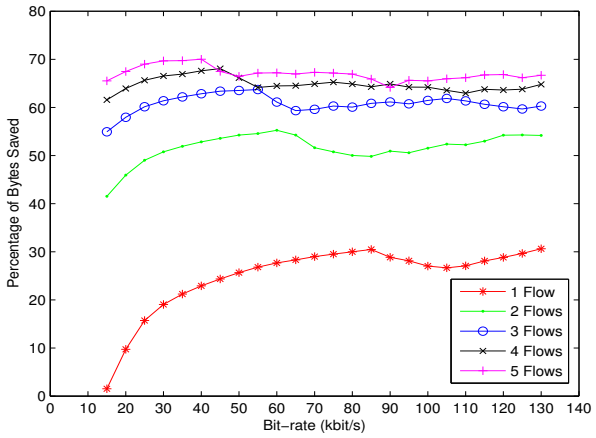

Figure 8. Percentage of bytes saved due to flow aggregation in LT & LD applications.

stay constants (we can achieve a maximum savings of around 90%). In Figure 7, we also observe that savings achieved by WiOptiMo increase as the number of application flows increases.

Finally, for applications with low throughput and low delay requirements, we could achieve a maximum saving of 70% (see Figure 8). Even at very low bit-rate (∼20kbit/s), WiOptiMo is able to save 10% of the data transferred between client and server.

B. QoS support by WiOptiMo

To test the capability of the WiOptiMo with an aggregation schema to provide QoS support, we setup a wireless mesh network testbed composed by three static Internet-sharing nodes and two wireless mobile nodes. Each static node consists of an ALIX.2D2 system board, which supports two mini-PCI radios. We used one Wistron DNMA92 miniPCI card for each board, which is in turn connected to two 802.11n antennas. Each board mounts a 500 MHz AMD Geode LX800 processor and 256 MB DDR DRAM, runs Debian Wheezy 7.0 with Linux Kernel 3.12.6, and uses an ath9k driver for WiFi. We used two ASUS EeePC 900 (with an Atheros 5008 Wireless Card, a 900MHz Celeron Processor and 1GB DDR RAM) as mobile nodes in our experiments. They operated on Debian Wheezy 7.0 with an ath5k WiFi driver.

We utilized Iperf and measured the throughput between client and server using two different flow classes (HT & LD and HT & HD), in two distinct configurations: with a single SNAPT and with two SNAPTs.

To complete the hardware setup, we installed WiOptiMo SNAPT on two Dell Optiplex 760 (servers) and a Lenovo ThinkPad T410a had WiOptiMo CNAPT installed on it. Both machines operated on a Linux distribution (Ubuntu 12.04 with Linux kernel 3.11). Two static nodes (gateways, A and C) and two servers were connected to the Internet with an Ethernet connection, while the rest of the nodes (A) participate in the mesh network. We set the bandwidth of Ethernet connection to 10Mbit/s. The gateways performed NAT between the mesh network and the Internet. We ran the Optimised Link State Routing Protocol daemon (OLSRd, version 0.6.2) [29] on each node for network path resolution and configured the network to ensure that the two SNAPTs could be reached by separate gateways. The final testbed architecture is shown in Figure 9.

Results show that a software configuration with multiple SNAPTs increases the network throughput and then helps preserving the QoS of applications. This is clearly visible in Figure 10, which illustrates the throughput comparison in a single SNAPT and in a double SNAPT (with different network delays and accessible from separate gateways) configuration. In the first scenario, the available bandwidth gets divided equally between the two application classes. In the second scenario, the HT & HD class achieves on average higher throughput compared to HT & LD class because the data of HT & LD class always gets routed to the SNAPT with lowest delay. Specifically, in the two SNAPT scenario, we observe a higher throughput compared to the bandwidth available towards each single gateway. Finally, we did not observe any significant additional delay in the network due to the introduction of WiOptiMo.

VII. MIPv6 COMPARISON

To assess the performance of our WiOptiMo framework with respect to SoA protocols, we compared the behavior of WiOptiMo and MIPv6 [7], which is the standard proposed by IETF to handle mobility of Internet hosts for mobile data communication in IPv6 networks. We focused on IPv6 since it is the basis of the future All-IP networks, as it can be seen for example with the 3GPP decision of adopting IPv6 as the only IP version for an IMS.

The adaptation of our WiOptiMo framework for IPv6 networks was straightforward: the sockets used by WiOptiMo for communications were upgraded to use both IPv4 and IPv6, while a patch, specifically developed for this test, was added to the framework to ensure the exclusive use of the IPv6 protocol. Internal data structures were already designed to store and process IPv6 traffic, so no further modifications were needed.

We measured the following performance parameters:

1) *Handoff latency*. Defined as the time interval between the last data segment received through the old path and the first data segment received through the new path from mobile host to corresponding node (CN).
2) *Packet loss rate*. Defined as the number of lost packets due to handover divided by the total number of packets sent by the CN.

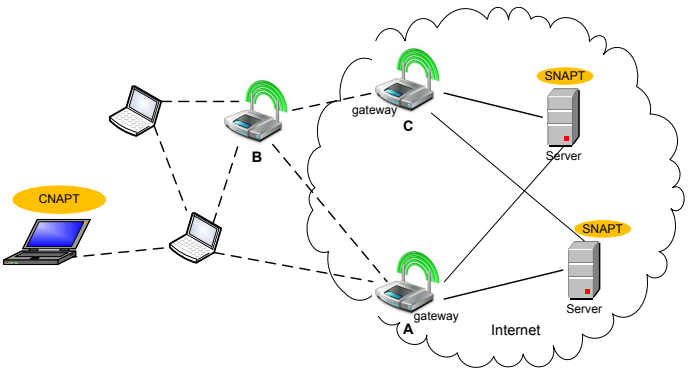

Figure 9. Testbed mesh network architecture.

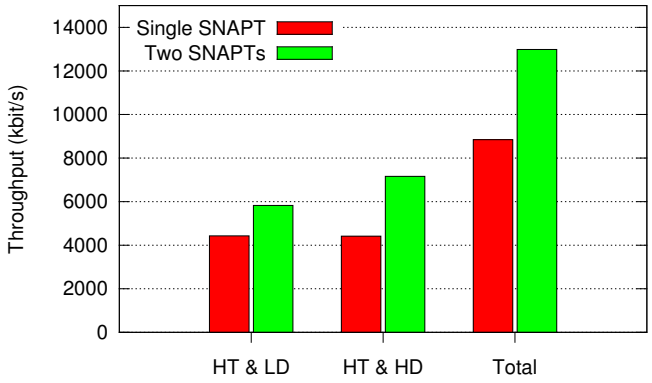

Figure 10. Throughput with multiple SNAPTs.

3) *Throughput*. Defined as the total useful bits that can be delivered to the mobile host upper layer application, divided by the time (estimate of the average transmission speed).

The baseline for the performance comparison is a standard IPv6 network configuration, without any mechanism for mobility support.

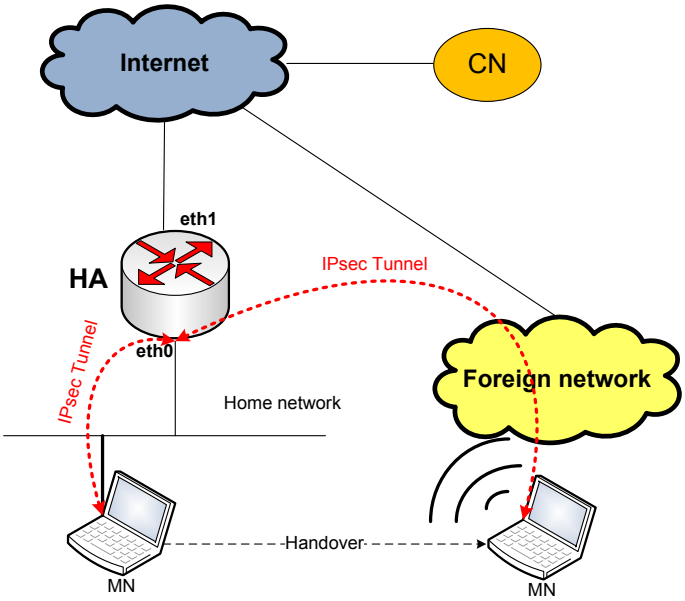

Figure 11. MIPv6 testbed setup

A. MIPv6 Testbed description

We setup a Mobile IPv6 testbed with IPsec static keying, using the UMIP open source implementation for Linux [30]. UMIP supports the IETF RFC RFC6275 (Mobile IPv6)[7]. Figure 11 shows the main elements of the testbed setup, which are:

- The **Home Agent** (HA). Its egress interface (*eth1*) is connected to the Internet, while its *etho* interface is connected to the home link of the mobile node.
- The **Mobile Node** (MN). The MN is initially connected to its Home network using its ethernet interface. Then, after a handover, it connects to a foreign network using its WiFi interface. The traffic exchanged between the MN and its HA is IPsec protected (tunnel mode).

We installed the HA on a Dell Optiplex 760 computer running a Debian 7.6 (with Linux kernel v. 3.14-2) distribution. The MN we used for our experiments was a HP Folio i3 laptop, running Debian 7.6, equipped with an 802.11b/g/n WiFi card.

In the standard MIPv6 configuration, the communication between the MN and the CN is routed through the HA. To enhance the performance of MIPv6, the Route Optimization (RO) protocol was introduced. The RO enables a MN and a CN to communicate directly, bypassing completely the HA. The RO technique works in this way: after a handover, when the MN receives the first tunneled message from the HA, the MN informs the CN about its new *care-of-address*, by sending a Binding Update (BU) message. The CN stores the home address plus care-of-address into its Binding Cache. Once the new entry is stored, communication directly takes place between MN and CN. To make RO secure, an authentication and encryption mechanism between MN and CN must be set up. The current MIPv6 specification defines that Return Routability (RR) [7] should be used for authentication between MN and CN. The RR procedure assumes that a CN has a private key and a random number that it renews at regular intervals. Although the RR procedure can be easily setup in a laboratory testbed, it is unlikely that every CN is configured for accepting our public certificate. Furthermore, setting up the RR mechanism in every CN is costly. Actually, none of the IPv6-enabled hosts in the 100 Top Internet Websites [31] are configured to be a CN. For this reason, we setup our MIPv6 testbed without RO.

B. WiOptiMo Testbed description

We setup a testbed with WiOptiMo in a single SNAPT configuration scenario. We installed WiOptiMo SNAPT on a Dell Optiplex 760 (server), running a Linux distribution (Ubuntu 14.04). WiOptiMo CNAPT was installed on the HP Folio i3 laptop.

We tested the performance of WiOptiMo in the scenario of WiFi micro-mobility (i.e., handover between WiFi access points of the same provider). We simulated a MN moving between the coverage area of two 802.11 access points with different SSID and IP networks, by manually switching the connection to the access point with the *wpa_gui* tool, a GUI interface for *wpa_supplicant* [32] that enables a user to choose which configured network to connect to.

C. Results

To measure the handover latency, we first connected our MN to an 802.11n access point and then induced an IP and gateway change in the WiFi network, by manually connecting to a new 802.11 access point. As a consequence, the connection was re-routed through a new gateway. To be deterministic, we did not use DHCP to get the IP address of the MN, but used a static IP network configuration. In typical WMNs without any mobility support mechanism, the change of gateway implies the change of external IP address and the need of re-establishing the connection. We used WiOptiMo and MIPv6, and tested their capability to spot the route change and preserve an ongoing transport session.

In WiOptiMo, handover latency is affected by the protocol used by the application. If TCP protocol is used, latency depends on the timeout used by WiOptiMo to detect a broken link or a network traffic stall. Since we wanted to provide absolute values for latency, we measured it by running the standard *ping* utility, with the flood (*-f*) option. In the flood ping, for every $ECHO\_REQUEST$ sent a period "." is printed, while for every $ECHO\_REPLY$ received a backspace is printed. This provides a rapid display of how many packets are being dropped. Since we did not specify any *interval* seconds between sending each packet, packets were transmitted without waiting. Session length of ICMP packets was 60s, packet size was 84 bytes, packets were output as fast as they came back or at one hundred times per second (whichever is more). During the test, we switched the connection between the access points so that the client's ICMP connection was re-routed to a different gateway. At the server side we logged all the socket connections received by SNAPT. We registered the handover latency for WiOptiMO by measuring the time it took for SNAPT to receive ICMP packets (generated by ping at client side) from the new gateway. We iterated this experiment for 50 times.

To measure the handover latency for MIPv6, we transmitted ICMP packets between the MN and the server where we previously run the SNAPT, and computed the difference between the timestamp of the last ICMP packet from the old gateway and the first ICMP packet from the new gateway.

To understand the impact of the overhead introduced by the mechanisms used by WiOptiMo and MIPv6 to manage handover, we recorded the time to complete a layer 3 switch without any mobility support mechanism. This time comprises the following components:

- Disassociation/deauthentication from the current access point.
- Authentication/association to the new access point.
- WPA key negotiation.
- Static IP address loading.

To measure the degradation on throughput and the additional end-to-end delay introduced by WiOptiMo and MIPv6, we used the *netperf* [33] (version 2.6.0) benchmark utility. Netperf is composed of a client and a server (*netserver*) application. The client was installed and run on the MN. It takes as input the IP address of the server, the server port number for TCP control connection and the TCP packet size (bytes). Each experiment lasted for 10 seconds and the netperf client application gave as output the observed throughput (in kbit/s) and the end-to-end delay (in msec). The socket buffer sizes at client (send buffer) and server (receive buffer) were set to their default standard values in Linux. The default TCP send and receive buffer size was 16,384 bytes and 87,380 bytes respectively. The size of each packet transmitted by the client was the same as the send buffer size (i.e., 16,384 bytes). The netserver installed at the server side listened at port number 12865 (default value) for control connections initiated by the client. Degradation on throughput and additional end-to-end delay were measured by running the netperf client application a) without any mobility framework, b) with a MIPv6 setup and c) with WiOptiMo running in background.

Handoff latency: Figure 12 reports the normalized probability density function of the time required by WiOptiMo and MIPv6 to perform the overall handover process when the MN connects to a new access point with a different gateway, while downloading a file via HTTP. The mean of the latency time for WiOptiMo and MIPv6 is shown in Table III. As it can be seen, mean handover latency time is similar for the two solutions, but WiOptiMo slightly outperforms MIPv6. The impact of the mechanisms for managing handover on latency is clearly visible by looking at the mean time required by the operating system for performing a layer 3 switch between the two different access points. It can be observed that mechanisms for managing handover accounts for about 2/3 of the latency time.

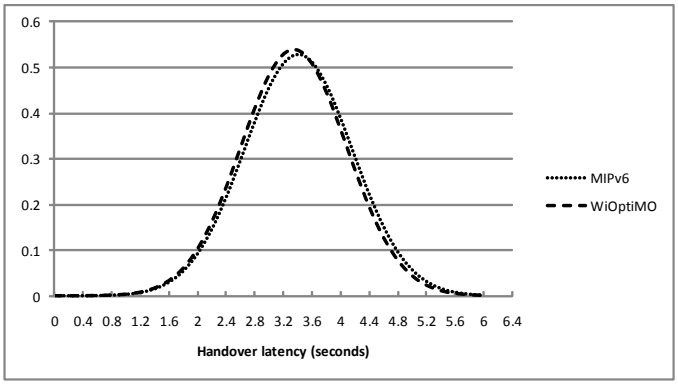

Figure 12. Probability density function of handover latency with no DHCP protocol

TABLE III. Measured Mean Handoff Latency

	Handoff Latency
Without mobility support	1.20 s
WiOptiMo	3.34 s
MIPv6	3.40 s

Throughput: The set of measurements on throughput show that the degradation introduced by WiOptiMo is acceptable and that the throughput experienced in WiOptiMo outperforms (more than six orders of magnitude better) the throughput experienced in MIPv6 (Table IV).

WiOptimo reduces throughput only by less than 7%: this is mainly due to the computational overhead that is needed for capturing packets at the CNAPT side and signing them, and for checking their integrity at the SNAPT side (and doing the reverse). In standard MIPv6, the drop in throughput is significant (more than 41%) and mainly due to the lack of route optimization: instead of sending packets directly to the MN, the correspondent node sends packets to the MN's home address, which will then encapsulates and forward them to the MN. This mechanism clearly decreases the throughput.

End-to-end delay: To measure end-to-end delay, we used the TCP request/response test of netperf. The request/response performance test consists in executing a transaction, which includes the exchange of a single request and a single response of given sizes. From a transaction rate, the one way and round-trip average latency can be inferred. The TCP request/response

TABLE IV. Measured Throughput

	Throughput
Baseline	50.09 Mbits/s
With WiOptimo	46.77 Mbits/s
With MIPv6	29.45 Mbits/s

test can be invoked with netperf using the $-t$ option with TCP_RR as argument, and the $-r$ option to set the request and/or response sizes.

Table V reports the results for the end-to-end delay, obtained by running the netperf utility when the TCP packet size varies from 1 byte to 10^8 bytes. The client was connected through a 802.11n network to the netserver server. As it can be seen from measurements, under the same network conditions (default TCP send and receive buffer size set to 16,384 bytes and 87,380 bytes respectively), end-to-end delay depends on the packet size (it increases as the packet size increases). Both WiOptiMo and MIPv6 have worst performances compared to a network configuration without mobility frameworks. WiOptiMo always outperforms (from about 10 to about 3 orders of magnitude) MIPv6. In comparison with the baseline configuration, the performance degradation of MIPv6 is noticeable (up to 13 orders of magnitude) for small packet sizes (1 to 10^4 bytes), while the overhead introduced by WiOptiMo is on average only 1.2 orders of magnitude. Finally, the degradation in percentage on end-to-end delay introduced by WiOptiMo is smaller for large packet sizes. In a typical video streaming usage scenario, which involves large application packet sizes, WiOptiMo has a minor impact on end-to-end delay performance.

VIII. CARMNET USAGE SCENARIO

We demonstrated that WiOptiMo has better performance than MIPv6 in terms of handover latency, packet loss rate and throughput. In this section, we also show its application in other contexts. In particular, we present a scenario where WiOptiMo is employed to support wireless network coverage extension by its integration with a resource allocation framework called CARMNET. Furthermore, we show how the amount of bytes saved thanks to the WiOptiMo aggregation scheme can be taken into account in the computation of a utility-based resource allocation policy.

A. CARMNET architecture

The idea of the CARMNET system was proposed in [4]. CARMNET enables its end users to share their resources, in particular to share the Internet access. The system consists of multiple components deployed both on a client- and server-side. The CARMNET overall system architecture is illustrated in Figure 13.

The DANUMS Loadable Kernel Module (LKM) [34] is an implementation of the Delay Aware Network Utility Maximization (DANUM) model developed for the Linux environment. DANUM is an optimization framework for wireless multi-hop networks that incorporates the delay factor into the computation of network's flows utility connected to the Network Utility Maximization (NUM) model [35]. The subsystem works in the kernel space, which allows for an integration with the network stack necessary to introduce new queuing and scheduling mechanisms. The OLSR daemon, a popular

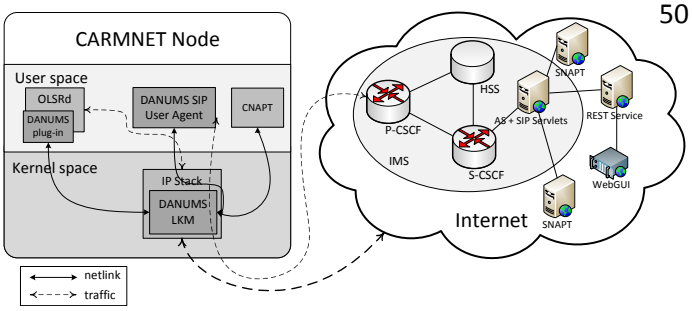

Figure 13. Architecture of the CARMNET system.

implementation of the OLSR routing protocol, is used for the network path resolution and signaling the DANUM-specific information in a distributed way. To ensure a communication between both subsystems, deployed in both the kernel and user space respectively, the Netlink protocol is applied. The Netlink protocol is used to communicate with the client-side part of the CARMNET mobility subsystem based on the WiOptiMo framework. The mobility services are provided by means of the client proxy - CNAPT - installed on the CARMNET wireless node. The role of the component is to intercept traffic flows associated to the mobility service and relay them to the SNAPT server. To ensure scalability and avoid concentrating traffic flows in one single spot, multiple SNAPTs are located on the Internet.

The last subsystem employed on a CARMNET Wireless node is a DANUMS SIP User Agent, which is responsible for exchanging information with an IMS architecture. This integration allows the CARMNET system to use an enhanced IMS infrastructure to provide the session and user management and exploit the unique features of an utility-aware flow control and resource allocation (provided by DANUMS [34]). An user of the CARMNET system can review and/or modify its information via the web application WebGUI integrated through REST service with the IMS infrastructure. One of the goals of the CARMNET system is to make the telecom operator IMS services effectively available to users of WMNs. As a result of integrating the carrier-grade AAA with the NUM-oriented resource management , the system enables the application of utility-based charging based on the *denarii* (i.e., the virtual units of utility) unit of the DANUM subsystem, which may be used as a market-like regulator for utility- and reliability-oriented resource allocation.

The CARMNET system goals include providing an access to the Internet in places without (or with very weak) WiFi signal from the static infrastructure. It may be particularly useful in metropolitan networks, where extending range by means of a static infrastructure can be expensive. In contrast, the CARMNET system, as a distributed and dynamic solution based on the wireless mesh networking paradigm, employs wireless nodes to extend the range of a network.

B. Seamless Horizontal Handover

As presented in [5], there exist at least two scenarios where CARMNET-based solutions can be employed.

The first scenario is based on the network coverage extension concept (see Figure 14). The most of deployed wireless networks incorporate only static infrastructure to provide their services. The infrastructure is very often cheap in maintenance,

TABLE V. End-to-end delay

Packet size [bytes]	1	10	10^2	10^3	10^4	10^5	10^6
Delay baseline [milli sec]	0.641	0.653	0.662	0.803	2.112	14.490	134.804
Delay with WiOptimo [milli sec]	0.794	0.799	0.804	0.891	2.331	16.257	156.235
Delay with MIPv6 [milli sec]	8.18	8.235	8.87	10.246	14.02	54.919	407.381
WiOptiMo Degradation [%]	19.27	18.24	19.29	11.5	9.42	10.87	13.71
MIPv6 Degradation [%]	92.16	92.06	92.53	92.15	84.94	73.62	66.9

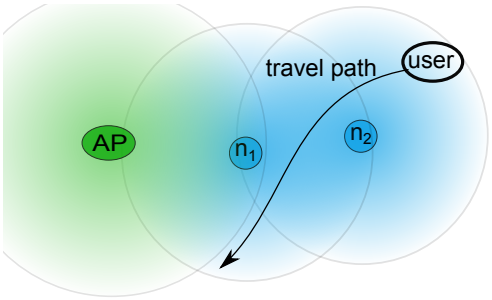

Figure 14. Example network topology in network coverage extension scenario.

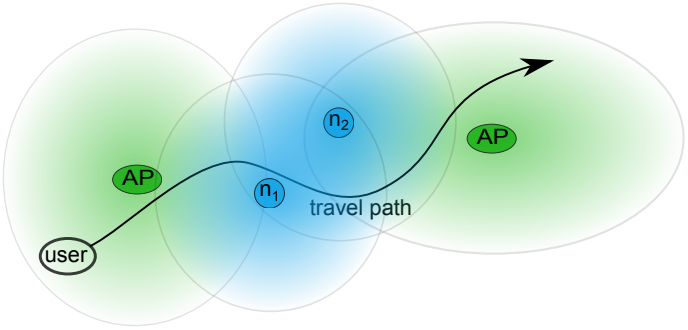

Figure 15. Network topology in handover scenario.

however, it covers only a small area and an extension of its range is expensive. The proposed scenario allows to reduce this cost by shifting it to the users of the network, as the result of adopting the WMN paradigm. This approach benefits from the distributed locations of users - who are rarely concentrated in one place - that leads to quite large overall coverage. To put it into practice, each user device must run the CARMNET client software, which enables sharing Internet access with other users. As described in the previous section, the client software is not only responsible for sharing a network access, but also introduces a resource management system that takes care of the users' willingness to serve their own or other users' traffic.

However, an important issue is how to encourage users to share resources on their devices (e.g., CPU load, battery, bandwidth) and a network access. In this respect, CARMNET defines a virtual currency based on a utility function derived from the NUM optimization problem. The utility function depends on traffic parameters like throughput and delay. In the CARMNET system, the *denarii* virtual currency is utilized to charge users. Furthermore, it enables incorporating an incentive model which may enforce collaboration between users to effectively use the network.

A more complex scenario than the network range extension is the provision of the Internet access by multiple independent infrastructures and wireless networks in a single broad area (see Figure 15). In this scenario, a CARMNET infrastructure can be set up in order to bridge the gap between two (or more) access points. As most users are mobile and use Internet on the move (e.g., they perform a videocall or download some documents), a transition between independent networks can be an issue. In order to solve this problem, there is a need to provide handover services between CARMNET-based networks and the independent wireless networks, which will enable the possibility to transparently transfer ongoing sessions between networks without interruption. In the CARMNET system, the role of the handover service provider is played by the WiOptiMo framework, which is integrated with the rest of the components.

C. The Usage of Virtual Currency

Incentive mechanisms play an important role for encouraging users to use CARMNET-based networks. An incentive mechanism can be defined based on the concept of virtual currency introduced and used by the DANUM framework [34], which was primarily applied only for flow management purposes. The DANUM framework, as the application of the Delay-Aware variant of the Network Utility Maximization framework, determines utility of each served flow according to the flows' parameters like delay and throughput. Then, the value of the virtual currency rate y_f of flow f is calculated as a solution of the primal DANUM problem:

$$\max_{y_f} \sum_f U_f\left(y_f\right), \tag{1}$$

subject to the constraints associated with the system of virtual queues (see [34] for detailed description of the Delay-Aware NUM model). The $U_f(y_f)$ function is an utility function defined to represent characteristics of the flow f, i.e., according to the type of service. Examples of utility functions for TCP and UDP protocols are defined in [34].

As a main step of the DANUM flow control mechanism, a virtual unit value is calculated as derivative of the flow's utility. This value is then used to build virtual queues (managed in parallel with the real packet queues). The levels of virtual queues are thereafter used to schedule flows [34].

In [5], a model of the rewarding mechanism was proposed, however, it has been provided only for traffic forwarded inside each CARMNET network. In the scenario of seamless handover between multiple networks (based on WiOptiMo), such a simplified model is not sufficient. As described in Section V, WiOptiMo aggregates multiple flows belonging to the same class into one UDP flow. This approach optimizes the utilization of bandwidth, however, it compromises the capability of the DANUM subsystem to charge users, since flows are aggregated and DANUM is not able to compute the accurate number of virtual units.

Because of the incorrect utility estimation, the CARMNET users' account balance can be charged in a wrong way. For this reason, we propose a modification to the existing charging algorithm, which involves extending the functionality of the SNAPT component by adding an additional reporting mechanism. Since the SNAPT knows exactly the delay and throughput of each aggregated flow, it reports this information to the CARMNET IMS subsystem, which in turn uses it to recalculate the users' traffic reports and the account balance.

At the source node of each flow, the price of the virtual unit is computed using the standard partial derivative formula as follows:

$$\begin{aligned} U'_f(y_f(t)) &= \frac{\partial U}{\partial y}(y_f(t), \mathbf{v}) \\ &= \frac{\partial U}{\partial x}(x(y_f(t), \mathbf{v}), d(y_f(t), \mathbf{v}))\frac{\partial x}{\partial y}(y_f(t), \mathbf{v}) \\ &+ \frac{\partial U}{\partial d}(x(y_f(t), \mathbf{v}), d(y_f(t), \mathbf{v}))\frac{\partial d}{\partial y}(y_f(t), \mathbf{v}), \end{aligned} \tag{2}$$

where x and d denotes the end-to-end throughput and delay, respectively, and the exact values of utility derivatives are calculated using the linear interpolation. The values of the derivatives of x and d with respect to y are assumed to be constant [34] and set experimentally. The vector $\mathbf{v}$ represents other than delay and throughput flow parameters, which may affect user-perceived utility for a given flow, e.g., the jitter or packet loss. Formula 2 is used for offline price recalculations (according to reported values of delay and throughput) of served and aggregated flows regarding the real utility. In the next step, the new price is used to update users' account balance to the appropriate value.

In other words, although seamless handover provided by WiOptiMo is an important feature, it is much harder to correctly estimate flow utility in the mobility scenario. This might lead to the situation where the DANUM resource management system under- or overestimates the utility of traffic and, as a result, to the decrease of the service quality. To address this issue, we have introduced a rewarding mechanism for CARMNET users who utilize the WiOptiMo mobility service. This mechanism involves a discount in the virtual currency that is proportional to the amount of traffic served by WiOptiMo (reported to the IMS subsystem by SNAPT). This way, we acknowledge the benefit for the CARMNET network derived from the amount of bytes saved thanks to the use of the WiOptiMo aggregation scheme.

IX. Conclusion

In this paper, we have addressed the issue of supporting QoS expectations of mobile users in a wireless mesh networking environment and proposed a flow classification and aggregation scheme based on the WiOptiMo framework, to manage multiple applications with different QoS requirements. We evaluated our scheme on a Linux-based wireless mesh network testbed and showed that the aggregation mechanism improves network performance in terms of link utilization and QoS, while still providing mobility support. We also tested WiOptiMo in a IPv6 network and demonstrated that it outperforms the Mobile IPv6 protocol in terms of handover latency, packet loss rate and throughput. Finally, we have proposed an incentive mechanism for motivating nodes that utilize the WiOptiMo mobility framework to share their network resources with other nodes of a wireless mesh network. The strengths of our framework are that it does not require any changes to be made to the network protocol stacks of either the mobile or fixed end systems, it does not suffer from the scalability problems of Mobile IPv6 because it enables an efficient management of local mobility, and it can be easily integrated into a utility-based resource allocation framework.

Acknowledgment

This work is supported by a grant from Switzerland through the Swiss Contribution to the enlarged European Union (PSPB-146/2010, CARMNET).

References

[1] D. Gallucci, S. Mudda, S. Vanini, and R. Szalski, "A Flow Aggregation Scheme for Seamless QoS Mobility Support in Wireless Mesh Networks," in *MOBILITY 2014, The Fourth International Conference on Mobile Services, Resources, and Users*, Paris, France, July 2014, pp. 32–37.

[2] S. Jakubczak, D. G. Andersen, M. Kaminsky, K. Papagiannaki, and S. Seshan, "Link-alike: using wireless to share network resources in a neighborhood," vol. 12, no. 4. ACM, 2009, pp. 1–14.

[3] C. Middleton and A. B. Potter, "Is it good to share? A case study of FON and Meraki approaches to broadband provision," in *Proceedings of International Telecommunications Society 17th Biennial Conference*, Montreal, 2008.

[4] M. Glabowski and A. Szwabe, "Carrier-Grade Internet Access Sharing in Wireless Mesh Networks: the Vision of the CARMNET Project," in *AICT 2013, The Ninth Advanced International Conference on Telecommunications*, June 2013, pp. 113–116.

[5] P. Walkowiak, R. Szalski, S. Vanini, and A. Walt, "Integrating CARMNET System with Public Wireless Networks," *ICN 2014, The Thirteenth International Conference on Networks*, pp. 172–177, February 2014.

[6] C. Perkins, "IP Mobility Support for IPv4, Revised," RFC 5944 (Proposed Standard), Internet Engineering Task Force, 2010, last retrieved: 06.07.2015. [Online]. Available: http://www.ietf.org/rfc/rfc5944.txt

[7] C. Perkins, D. Johnson, and J. Arkko, "Mobility Support in IPv6," RFC 6275 (Proposed Standard), Internet Engineering Task Force, Jul. 2011, last retrieved: 06.07.2015. [Online]. Available: http://www.ietf.org/rfc/rfc6275.txt

[8] R. Koodli, "Mobile IPv6 Fast Handovers," RFC 5568 (Proposed Standard), Internet Engineering Task Force, 2009, updated by RFC 7411. Last retrieved: 06.07.2015. [Online]. Available: http://www.ietf.org/rfc/rfc5568.txt

[9] H. Soliman, C. Castelluccia, K. ElMalki, and L. Bellier, "Hierarchical Mobile IPv6 (HMIPv6) Mobility Management," RFC 5380 (Proposed Standard), Internet Engineering Task Force, Oct. 2008, last retrieved: 06.07.2015. [Online]. Available: http://www.ietf.org/rfc/rfc5380.txt

[10] G. A. Di Caro *et al.*, "Deployable Application Layer Solution for Seamless Mobility Across Heterogeneous Networks." *Ad Hoc & Sensor Wireless Networks*, vol. 4, no. 1-2, pp. 1–42, 2007.

[11] T. Ernst, "Network Mobility Support Goals and Requirements," RFC 4886 (Informational), Internet Engineering Task Force, 2007, last retrieved: 06.07.2015. [Online]. Available: http://www.ietf.org/rfc/rfc4886.txt

[12] K. Ahmavaara, H. Haverinen, and R. Pichna, "Interworking architecture between 3GPP and WLAN systems," *Communications Magazine, IEEE*, vol. 41, no. 11, pp. 74–81, 2003.

[13] 3GPP Specifications. Last retrieved: 06.07.2015. [Online]. Available: http://www.3gpp.org/specifications/

[14] M. Buddhikot, A. Hari, K. Singh, and S. Miller, "MobileNAT: A New Technique for Mobility Across Heterogeneous Address Spaces," *ACM Mobile Networks and Applications*, vol. 10, no. 3, pp. 289–302, 2005.

[15] I. Ramani and S. Savage, "SyncScan: Practical Fast Handoff for 802.11 Infrastructure Networks," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 1, 2005, pp. 675–684.

[16] V. Navda, A. Kashyap, and S. R. Das, "Design and Evaluation of iMesh: an Infrastructure-mode Wireless Mesh Network," in *World of Wireless Mobile and Multimedia Networks. WoWMoM 2005*, Italy, June 2005, pp. 164–170.

[17] Y. He and D. Perkins, "BASH: A Backhaul-Aided Seamless Handoff Scheme for Wireless Mesh Networks," in *World of Wireless, Mobile and Multimedia Networks. WoWMoM 2008*. IEEE, June 2008, pp. 1–8.

[18] R. Huang, C. Zhang, and Y. Fang, "A Mobility Management Scheme for Wireless Mesh Networks," in *Global Telecommunications Conference. GLOBECOM'07. IEEE*, Washington DC, USA, November 2007, pp. 5092–5096.

[19] M. Sabeur, G. Al Sukkar, B. Jouaber, D. Zeghlache, and H. Afifi, "Mobile Party: A Mobility Management Solution for Wireless Mesh Network," in *Wireless and Mobile Computing, Networking and Communications. WiMOB 2007*, October 2007, p. 45.

[20] S. Speicher and C. H. Cap, "Fast Layer 3 Handoffs in AODV-Based IEEE 802.11 Wireless Mesh Networks," in *Wireless Communication Systems. ISWCS'06*. IEEE, 2006, pp. 233–237.

[21] Y. Amir, C. Danilov, R. Musuăloiu-Elefteri, and N. Rivera, "The SMesh Wireless Mesh Network," *ACM Transactions on Computer Systems (TOCS)*, vol. 28, no. 3, pp. 6:1–6:49, September 2010.

[22] D. Gallucci, S. Giordano, D. Puccinelli, N. Tejawsi, and S. Vanini, "Fixed Mobile Convergence: The Quest for Seamless Mobility," in *Fixed/Mobile Convergence Handbook*. CRC Press, 2010, pp. 185–196.

[23] S. Vanini, D. Gallucci, S. Giordano, and A. Szwabe, "A Delay-aware NUM-driven Framework with Terminal-based Mobility Support for Heterogeneous Wireless Multi-hop Networks," in *IEICE Information and Communication Technology Forum 2013*, 2013.

[24] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," RFC 2104 (Informational), Internet Engineering Task Force, Feb. 1997, updated by RFC 6151. Last retrieved: 06.07.2015. [Online]. Available: http://www.ietf.org/rfc/rfc2104.txt

[25] IETF DiffServ Working Group page. Last retrieved: 06.07.2015. [Online]. Available: http://datatracker.ietf.org/wg/diffserv/charter

[26] R. Chakravorty, S. Katti, J. Crowcroft, and I. Pratt, "Flow Aggregation for Enhanced TCP over Wide-Area Wireless," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, 2003, pp. 1754–1764.

[27] iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool. Last retrieved: 06.07.2015. [Online]. Available: https://github.com/esnet/iperf

[28] dumpcap - The Wireshark Network Analyzer. Last retrieved: 06.07.2015. [Online]. Available: http://www.wireshark.org/docs/man-pages/dumpcap.html

[29] An ad-hoc wireless mesh routing daemon. Last retrieved: 06.07.2015. [Online]. Available: http://www.olsr.org

[30] UMIP - Mobile IPv6 and NEMO for Linux. Last retrieved: 06.07.2015. [Online]. Available: http://www.umip.org/

[31] List of most popular websites. Last retrieved: 06.07.2015. [Online]. Available: http://en.wikipedia.org/wiki/List_of_most_popular_websites

[32] Linux WPA/WPA2/IEEE 802.1X Supplicant. Last retrieved: 06.07.2015. [Online]. Available: http://w1.fi/wpa_supplicant/

[33] Netperf Homepage. Last retrieved: 06.07.2015. [Online]. Available: http://www.netperf.org/netperf/

[34] A. Szwabe, P. Misiorek, and P. Walkowiak, "Delay-Aware NUM System for Wireless Multi-hop Networks," in *Wireless Conference 2011-Sustainable Wireless Technologies (European Wireless), 11th European*, Vienna, Austria, April 2011, pp. 530–537.

[35] F. P. Kelly, A. K. Maulloo, and D. K. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research society*, vol. 49, no. 3, pp. 237–252, 1998.