Cominardi, Luca; Giust, Fabio; Bernardos, Carlos J.; Oliva, Antonio de la. Distributed mobility management solutions for next mobile network architectures. *Computer networks,* 121 (2017), pp. 124–136

DOI: https://doi.org/10.1016/j.comnet.2017.04.008

Distributed mobility management solutions for next mobile network architectures

Luca Cominardi[a,b,*], Fabio Giust[c], Carlos J. Bernardos[b], Antonio De La Oliva[b]

[a] IMDEA Networks Institute, Av. de la Universidad, 30, Leganés, Madrid, Spain
[b] University Carlos III of Madrid, Av. de la Universidad, 30, Leganés, Madrid, Spain
[c] NEC Laboratories Europe, Heidelberg, Germany

Keywords:
Distributed mobility management
Mobile networks
PMIPv6
SDN
Experimental evaluation

A B S T R A C T

The architecture of current operator infrastructures is being challenged by the non-stopping growing demand of data hungry services appearing every day. While currently deployed operator networks have been able to cope with traffic demands so far, the architectures for the 5th generation of mobile networks (5G) are expected to support unprecedented traffic loads while decreasing costs associated to the network deployment and operations. Distributed Mobility Management (DMM) helps going into this direction, by flattening the network, hence improving its scalability, and enabling local access to the Internet and other communication services, like mobile-edge clouds. Initial proposals have been based on extending existing IP mobility protocols, such as Mobile IPv6 and Proxy Mobile IPv6, but these need to further evolve to comply with the requirements of future networks, which include, among others, higher flexibility. Software Defined Networking (SDN) appears as a powerful tool for operators looking forward to increased flexibility and reduced costs. In this article, we first propose a Proxy Mobile IPv6 based DMM solution which serves as a baseline for exploring the evolution of DMM towards SDN, including the identification of DMM design principles and challenges. Based on this investigation, we propose a SDN-based DMM solution which is evaluated against our baseline from analytic and experimental viewpoints.

1. Introduction

Packet-based mobile networks have experienced a huge success in the last years, with the number of subscribers and traffic volume constantly growing. Several reports like [1] show that the mobile traffic growth will not decelerate, but increase 10-fold instead from 2015 by the end of 2020.

The envisioned scenario will not only assume a large data volume increase, but also a profound diversification of traffic and service demands, leading to a new environment for the telco industry which many operators have already identified as the 5th generation of mobile communications [2]. Since operators aim at improving their infrastructure to meet users' demands while reducing the associated deployment and operational costs, solutions providers have started looking at a wide plethora of aspects, as documented by the 3GPP's[1] New Services and Markets Technology Enablers (SMARTER) [3]. In addition, the telco industry is considering migrating towards a cloud-based infrastructure, adopting technologies like Software Defined Networking (SDN) and Network Function Virtualization (NFV) (see for instance the solution reported in [4]). Following these new technologies, the 3GPP's architectural study for next generation mobile systems, published in [5], focuses on enhancing the mobile network's core part, considering the evolution of the network towards a distributed and softwarized ecosystem.

Indeed, the current architecture of the 4G system is highly centralized and hierarchical, with control and data planes converged at the same packet gateway. Such entity both terminates the mobility signaling, and it forwards traffic to and from the mobile network enforcing the gating and policy functions. By doing so, the gateway acts as mobility anchor following the user movements by simply re-routing the packets over tunnels created with the access router where the user terminal is currently connected. But this simplicity comes with some penalties [6]: the mobility anchor represents a single point of failure, it poses scalability issues overloading the network core, and, in general, it leads to sub-optimal paths between the mobile nodes and their communication peers (also known as correspondent nodes, CNs).

Flattening the network architecture is regarded as one of the most promising approaches to design the architecture of the

* Corresponding author.
E-mail address: luca.cominardi@imdea.org (L. Cominardi).

[1] 3rd Generation Partnership Project, http://www.3gpp.org/.

next generation system, and the Distributed Mobility Management (DMM) paradigm goes precisely in such direction. Both 3GPP and IETF[2], which are the main standardization bodies in this area, have looked and are still looking at DMM-alike solutions. But, as highlighted before, there is also a very important tendency towards *softwarizing* wireless mobile networks, by adopting SDN and NFV approaches.

In this paper we first present a DMM solution based on a well known existing IP mobility protocol (Proxy Mobile IPv6), which could be referred to as *legacy* solution. Note that this solution has been designed by the authors of this work and contributed to the IETF [7,8]. This legacy solution is used as a baseline to identify the design principles and challenges of a DMM solution embracing the SDN paradigm, which could be in turn considered as *evolutionary* solution. Based on this analysis, we hence propose a new SDN-based DMM solution addressing the identified design principles and specific challenges. A unified methodology, which takes into account protocol specific operations (e.g., number and type of control messages, and mobility model), is proposed to analytically evaluate and compare both DMM solutions in terms of handover cost, scalability, and state space size. Finally, our proposed solutions are implemented in a test-bed leveraging commodity hardware and experimentally evaluated, with particular focus on the handover cost and its breakdown, providing insights on the different components of the overall handover latency and how they can affect network scalability.

The paper is organized as follows: Section 2 presents how the DMM concept started and how it is now evolving towards a SDN-based solution, which is then explained in more detail in Section 3. Section 4 provides a detailed mathematical analysis of selected metrics while Section 5 reports on experimental results. Finally, Section 6 presents a review of prior art and Section 7 concludes the paper.

2. DMM design considerations. Evolution from IP mobility towards SDN

The main proposition of DMM is simple: distributing mobility anchors by placing multiple ones closer to the location of the user. A lot of research has been conducted in this area, producing different kinds of solutions. The DMM Working Group (WG) at the IETF is one of the first and main venues where solutions for distributed IP mobility management are discussed. The group started exploring the DMM problem space by first looking at existing IP mobility protocols, like Mobile IPv6 (MIPv6) [9] and Proxy Mobile IPv6 (PMIPv6) [10]. The intention was to investigate possible extensions and adaptations to accepted standard protocols, in order to limit the impact on legacy implementations and equipment. Beyond IETF DMM and the already mentioned 3GPP's activity in [5], also the ONF's[3] Wireless and Mobile working group has taken its part, by proposing the adoption of SDN for the design of mobile networks.

As briefly introduced above, most of the first proposals of the DMM WG considered Proxy Mobile IPv6 as the baseline of the solution, as documented in [7,11,12]. PMIPv6 is one of the mobility protocols adopted by the 3GPP Evolved Packet Core (EPC) and provides network-based mobility whose main characteristic is to not require any active participations of the Mobile Node (MN) to support mobility. Worth noticing that the MN might be completely unaware of the Layer-3 mobility in place by the network. Other mobility protocols like MIPv6, or its DMM extension proposed in [13], provide instead a client-based mobility solution which requires the active involvement of the MN during attachment or handover procedures.

In the following section, we provide a detailed explanation on our solution for a PMIPv6-based DMM protocol, which has been contributed to the IETF in [7]. This solution is taken as basis for comparison since it represents many similar solutions extending existing mobility protocols and there is an analytic and experimental evaluation available [8].

2.1. IP mobility (PMIPv6) based DMM solution

The key entity in this solution is the Distributed Mobility Management Gateway (DMM-GW). The DMM-GW extends the PMIPv6 Mobile Access Gateway (MAG) functions incorporating most of the functionality of the PMIPv6 Local Mobility Anchor (LMA). Hence, a DMM-GW provides connectivity to IP based services, e.g., Internet, and has the capability of assigning and anchoring IPv6 prefixes. A unique IPv6 prefix pool belongs to each DMM-GW, from which a prefix is assigned to every MN attached to the DMM-GW's access links. In this way, a DMM-GW acts as a plain access router to forward packets to and from the Internet. Moreover, it is provided with mobility anchoring functions, that is, a DMM-GW is able to maintain the uplink and downlink forwarding for the IP flows that an MN started while attached to that DMM-GW, even after the MN has moved to a new DMM-GW. An external node, referred to as Control Mobility Database (CMD), is used to store the location of the MNs in the domain (i.e., the bindings).

In order to better understand the solution, we depict its main operations in Fig. 1, referred to with a number (#). First, a DMM-GW detects an MN attachment by using IPv6 Neighbor Discovery [14] signaling (1) (typically, an IPv6 host sends a Router Solicitation – RS – message upon joining a link) or by a dedicated link layer detection mechanism. Then, the DMM-GW notifies the CMD about the MN attachment by means of a Proxy Binding Update (PBU) message containing an IPv6 prefix reserved for the MN (2). In case of initial registration, there is no entry available in the CMD's cache for that particular MN, so the CMD registers for the first time the MN, by storing the IPv6 prefix assigned to the MN associated to the MN location, i.e., the DMM-GW's address. The CMD then acknowledges the operation to the DMM-GW with a Proxy Binding Acknowledgment (PBA) message (3), and the DMM-GW finally delegates the IPv6 prefix to the MN with a Router Advertisement message (4). Upon a handover, the messages (5,6) are sent, reflecting steps (1,2), but now the CMD receives a new IPv6 prefix in the PBU from the new DMM-GW (6), so the CMD associates the MN with the new prefix and the new location. The old location and prefix are included in a list of "anchoring" DMM-GWs and these parameters are conveyed to the new DMM-GW in the PBA message (7). In parallel, the CMD sends a PBU (8) to the "anchoring" DMM-GW including the parameters from the new DMM-GW, and then receives the PBA from the old DMM-GW (9). By doing so, both the new and old DMM-GWs have the necessary information to set up a tunnel between them to recover the ongoing IP flows. The tunnel is used for those flows started before the MN handed over from the previous DMM-GW, whereas new communications are handled by the new DMM-GW as a plain router, that is, without using any tunnels. This dynamic behavior is achieved by the MN obtaining a new IPv6 prefix from each DMM-GW it connects to (10). Consequently, an MN configures several IPv6 addresses, one per each visited DMM-GW, and its flows might be anchored at different DMM-GWs. One of the main advantages of this approach is that new flows started when the node is attached to the new DMM-GW are not tunnelled, hence they do not suffer from any overhead or non optimal routing, improving the overall performance of the network.

[2] Internet Engineering Task Force, http://www.ietf.org/.

[3] Open Networking Foundation, https://www.opennetworking.org/.

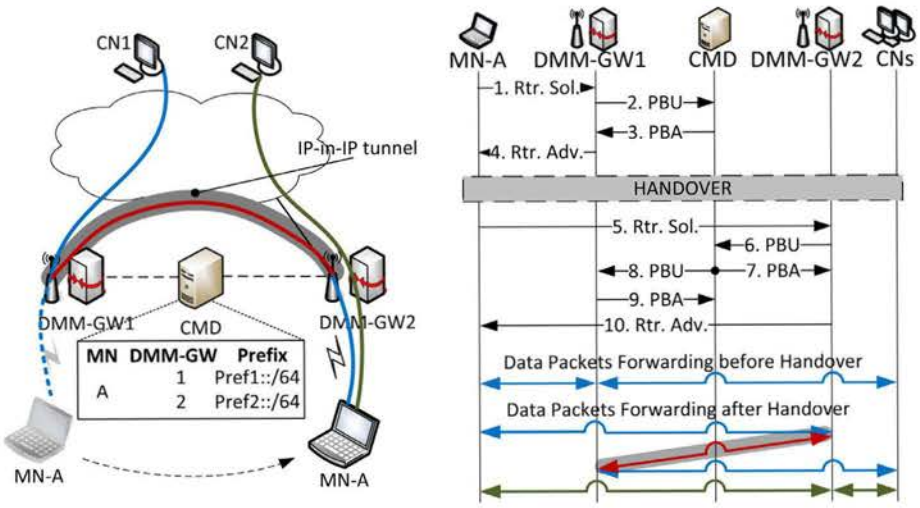


Fig. 1. PMIPv6-based DMM solution.

The above solution extends PMIPv6 to provide a flatter mobility architecture, and therefore, it is based on the same principles. Its main advantage is that it is an evolved solution based on existing mechanisms, that could be easily deployed on currently rolled-out networks. However, operators are already moving towards *software networks*, which are more flexible and allow for faster and richer service deployments.

2.2. DMM design principles, SDN challenges and opportunities

In this section we highlight the main components of a DMM solution and the challenges introduced by shifting the architecture paradigm from pure IP to SDN. The SDN concept separates the control and the data forwarding planes. Such separation allows for quicker provisioning and configuration of network connections. This approach decouples the system making decisions about routing (i.e., control plane) from the underlying system that forwards traffic to the selected destination (i.e., data plane). In an SDN environment, the entity in charge of implementing the control logic for the network is called Network Controller (NC) and it is responsible of configuring the nodes in the network (data plane) via a common application programming interface (API). A well-known SDN framework is the OpenFlow protocol and switch specification [15], which can be used by an external software application to program the data plane of network devices.

SDN enables the partitioning of the control system into modular parts that can be dynamically composed according to the network needs. Nonetheless, the choice of component partitioning can have a profound influence on the types of services ultimately delivered to the end user [16]. With this in mind, we depart from the DMM solution presented in the previous section and we try to answer at the following questions:

- *What are the tasks to be accomplished by a generic DMM solution to efficiently provide MNs with mobility support?*
- *What are the challenges and opportunities in accomplishing these tasks following the SDN paradigm?*

Generally speaking, each task can be seen as a stand-alone module that interacts with other modules in order to achieve MN's mobility. As a consequence, a generic DMM solution can be seen as a set of cooperating modules thus facilitating the evolution of such mobility solutions from IP architecture to SDN paradigm. These modules can be implemented and deployed in different ways. However, in order to provide a full-fledged mobility support to the MNs, they need to keep the same semantic interface towards the other modules. In the following we report the modules that a DMM solution should implement departing from our PMIPv6-based DMM solution:

- **Attachment detection.** As shown in Fig. 1, the whole PMIPv6-based DMM mobility procedure is triggered by a Router Solicitation upon MN attachment (1) or handover (5). By generalizing this concept, we can argue that a DMM solution requires a specific module capable of detecting the MN's attachment or handover. Moreover, such a module should be provisioned with the following information to effectively trigger the mobility procedure: *i)* the MN identity and, *ii)* the DMM-GW the MN is attached to. Moving towards SDN, such information can be also retrieved by other means that do not strictly belong to the IP solution space. For example, in addition to *Layer-3* mechanisms, the attachment (or handover) can be also detected at *Layer-2* or via a *dedicated interface*. For instance, LLC SNAP messages can be used for detecting new hosts in IEEE 802.1Q bridged networks while S1-MME dedicated interface can be used in 3GPP networks [17]. The possibility of employing different mechanisms, even simultaneously, however poses a significant challenge in the design of a SDN-based DMM solution: the MNs must be uniquely identified despite of the connectivity technology being used. In case of achieving such unique identification, a SDN-based DMM solution can potentially be extended across different technology domains thus providing an inter-technology mobility.
- **Binding.** Upon successful attachment detection in PMIPv6, the DMM-GW notifies the CMD about the IPv6 prefix reserved for the MN through PBU messages (2,6). In practical terms, we can postulate this operation as a stand-alone module providing a binding procedure that assigns one or more IPv6 prefixes to the MN. Moreover, the module keeps track of the assigned prefixes during the connection lifetime of the MNs. Therefore, we can identify two main tasks accomplished by this module: *i)* MN tracking and, *ii)* prefixes selection. The tracking function keeps record of the whereabouts of the MN along with the IPv6 prefixes assigned by the prefix selection function. While in IP based solutions the tracking function usually stores only the visited DMM GW, additional flexibility can be envisioned in SDN-based solution where supplementary localization services, e.g. GPS, could be leveraged. Such information can be

potentially taken into consideration next by the selection function which can select better anchor points for the MNs in combination with some network policies. From a deployment point of view, this function can be deployed in a *centralized* or *distributed* fashion. A *centralized* approach co-locates the selection and tracking functions, while a *distributed* approach separates them.

- **Prefix advertisement**. After the Binding function selects the prefix(es) to be assigned to the MN, the solution communicates them to the MN. For example, our PMIPv6-based DMM solution advertises such prefixes through Router Advertisement messages (4,10) [14]. Alternatively, such communication may also occur via DHCP Offer/DHCP Acknowledgement [18]. Upon RA reception, the MN configures its IP addresses starting from the assigned IP prefixes. It is worth highlighting that this paper addresses network-based DMM solutions, therefore no direct interaction is envisioned between the mobility protocol and the MN, which in turn can also be unaware of any mobility support. As a consequence, a DMM network-based SDN solution can only leverage *Layer-3* mechanisms for advertising the prefix. This function can be deployed in a *centralized* or in a *distributed* fashion, based on the entity in charge of communicating the prefix to the MN. In case of a *centralized* function, a single module (e.g., the network controller) takes care of retrieving all the MN's assigned prefixes and to communicate them to the MN via a unique message. On the contrary, a different implementation may rely on standard mechanisms to provide the assigned prefixes to the MN, for example allowing each assigned router to advertise its own prefix to the MN in a *distributed* way.
- **Traffic steering**. This module takes care of re-steering the MNs' traffic, thus effectively providing mobility. Upon an MN's attachment, the ongoing traffic must be re-routed to the new MN's location. This traffic path modification can be achieved in several ways, for instance, our PMIPv6-based DMM solution leverages on Layer-3 *tunnels*, i.e., IP-in-IP and/or the GPRS Tunneling Protocol [19]. An SDN-based solution can also envision the usage of other traffic steering techniques, such as VLAN/MPLS *tagging* [20,21], or *path reconfiguration*. This module may implement the whole function taking care of physically implementing it, or, it may interact with an external module or entity that is in charge of the configuration, e.g., a Path Computation Element (PCE) or a dedicated module on the NC. In the latter case, the traffic steering module communicates the paths that must be configured to the external module (i.e., a path from the MN to the DMM-GWs and vice-versa). Subsequently, the external module will configure the underlying network accordingly. It is worth highlighting that the SDN architecture defined in [16] enables SDN modules, like ours, to interact with the NC and request the configuration of some network paths. The NC has hence the responsibility to enforce the request in the underlying network, which may involve a combination of traffic steering techniques depending on the network deployment. For example, MN traffic steering can be a combination of Layer-3 tunnels and VLAN tagging across different network segments. Despite of the added complexity in the SDN approach, a careful combination of steering techniques can lead to a better usage of the resources due to a increased path optimality that can be potentially achieved in the network.

From the perspective of an SDN architecture, the modules described above can be implemented as applications running on top of one or multiple SDN controllers. The choice of using one or multiple SDN controller depends on the design adopted for the solution. So far we have only described the functionality required for a correct operation of a DMM solution. Nevertheless, the modules must cooperate and interact with each other. In order to do so, we need to the define the specific control and data planes used for the mobility solution[4]:

- **Mobility control plane.** It is the control plane adopted by the mobility solution and it can be *dedicated* or *compliant*. In case of a *dedicated* control plane, the mobility solution employs a different control plane for the signaling with respect to the one used by the SDN controller (e.g., based on PMIPv6 or GTP-C [22]). On the contrary, in case of a *compliant* control plane, the mobility solution uses the same southbound signaling used by the SDN controller (i.e., OpenFlow). A mobility solution may use a *mixed* control plane. For example, the attachment detection may employ the OpenFlow PacketIn event, while the communication between different modules leverages PBU/PBA messages.
- **Mobility data plane.** It can be *dedicated* or *shared*. A *dedicated* data plane implies that the packets exchanged between the MN and the DMM-GW (i.e., ARP, DHCP or IPv6 Neighbor Discovery) may follow a different path with regard to the packets belonging to the MN's data plane. In case of *shared* data plane, the two data planes are not separated.

We next describe in detail an SDN-based DMM solution, which will be later evaluated and compared to the PMIPv6-based one introduced before.

3. An SDN-based DMM solution

This section is devoted to the explanation of the proposed SDN-based DMM solution. As previously commented, designing an SDN-based DMM solution requires additional efforts with respect to a classic IP mobility solution where it is safe to assume that every node in the network speaks IP. However, this cannot be assumed in an SDN environment as introduced in the previous section. If we consider the more generic SDN concept – that is the decoupling of control and data planes – several boxes are controlled remotely by a Network Controller (NC) in order to accomplish a complex task in the network. In such scenario, the NC controls and instructs the data plane nodes via a Southbound interface which defines the instruction set understandable by both the NC and the data plane nodes. Although there are multiple possibilities for the choice of Southbound interface, we have decided to use the OpenFlow protocol, which enables the NC to write forwarding rules directly on the nodes. Given the nature of this paper, which evaluates the proposed DMM solutions from analytic and experimental viewpoints, our SDN-based mobility focuses on standard OpenFlow v1.5 capabilities, leaving potential extensions and non-standard Southbound interfaces out of the scope of this work. Undeniably, various Southbound-API may accomplish the same task in different ways but there are cases where one task cannot be accomplished by a specific Southbound-API.

For example, let's analyze the classical IP mobility mechanisms and OpenFlow. Classical IP mobility solutions (i.e., [9,10,23]) exploit IP tunnels for re-routing the traffic. Unfortunately, even the latest OpenFlow specification [15] does not include any instruction for managing IP tunnels[5]. Hence, in OpenFlow networks, mobility can be only supported by changing the forwarding rules in the data plane. According to the *Traffic steering* module described in Section 2.2, and to OpenFlow specifications, traffic steering can be only done via path reconfiguration or VLANs in an OpenFlow network.

[4] Note that these control and data planes refer only to mobility specific functionality and are different from the traditional control and data split considered in SDN approaches.

[5] Note there are some vendor specific extensions able to control some kinds of tunneling, e.g., GRE tunnels.

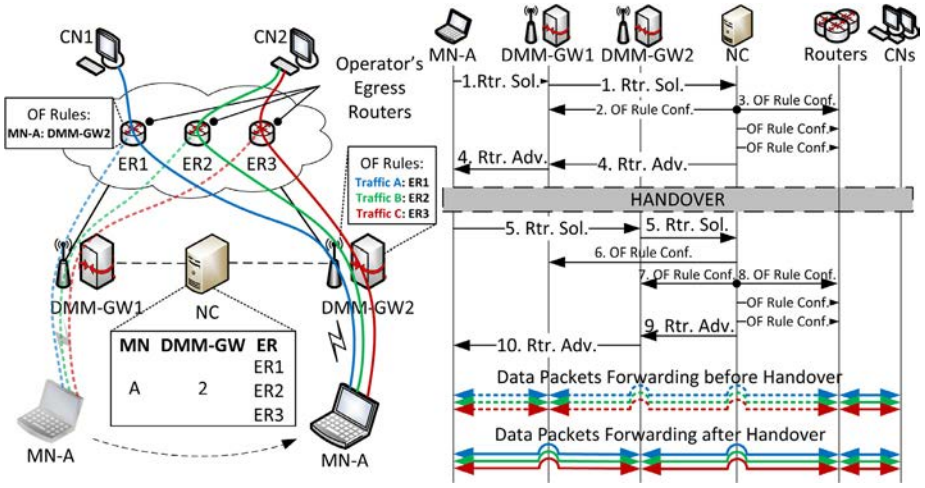


Fig. 2. SDN-based DMM solution.

In our proposed solution, the modules introduced when we discussed DMM design principles can be mapped to an SDN paradigm as different applications running on top of one or more NCs. Therefore, we designed the DMM-GWs as plain forwarding nodes (switches), bearing no mobility functionality, and hence delegating the whole intelligence to the NC. The applications adopt an event-driven communication paradigm in order to be as modular and reactive as possible, being this aspect fundamental in mobility solutions. Indeed, an event-driven communication can be used to fire triggers upon certain events, like the mobility support being activated by an MN handover.

As shown in Fig. 2, upon the attachment of an MN to an access point, the DMM-GW informs the NC, which assigns a network prefix (or a set of prefixes, in case differentiated treatment is required for fine-grained services) to the MN. The network prefix(es) is guaranteed to be unique by using a binding cache where the controller stores information about the MNs active in the network (and the prefixes they use). The detection of the attachment and the network prefix assignment in our solution, is based on IPv6 Neighbor Discovery as in the PMIPv6-based solution: the MN sends a Router Solicitation when attaches to the network (1,5), that serves as trigger, and the NC generates a Router Advertisement (RA) to communicate the network prefix(es) (4,10). These prefixes are anchored at a pool of k Egress Routers (ERs). After the selection and assignment of IP prefixes (and associated ERs), the NC configures the OpenFlow rules in the MN's target DMM-GW (2,7) and in the Egress Routers assigned to it (3,8). In case of handover, the NC also deletes the OpenFlow rules previously installed on the old DMM-GW (6).

Packet forwarding within the network is based on VLANs, which are statically pre-configured. Such VLAN paths connect Egress Routers with DMM-GWs. Note that these VLAN paths could also be dynamically configured by the NC using OpenFlow, but this procedure works in a different time scale with respect to mobility management and is ruled out of the scope of this work. As a matter of fact, packet re-routing could be also based on path reconfiguration. However, re-configuring the whole path leads to a higher and non-deterministic signaling load due to the variable length of the paths in the network.

Mobility support is achieved by installing OpenFlow rules at the Egress Routers and DMM-GWs, so packets destined or originated from an MN are tagged with the correct VLAN (see Fig. 2). Upon the attachment of an MN to the network, the NC configures Egress Routers to tag MN's packets with the VLAN connecting the Egress Router with the DMM-GW the MN is attached to. In case of handover, the NC simply needs to rewrite this rule at the Egress Router and the DMM-GW, selecting the correct VLAN that connects the new DMM-GW and the Egress Routers assigned to the MN.

Table 1
SDN-based DMM solution modules details.

Module	Depl.	Data req.	Events req.	Events prov.
Att. det.	L3	Router Sol.	OF PacketIn	MN attachment
Binding	Centr.	MN IDs	MN attachment	Binding up
Pref. Adv.	Centr.	Bind. cache	Binding up	-
Traf. steer.	VLAN	Bind. cache	Binding up	-

Summarizing, the solution envisions a shared data plane and a compliant control plane, exploiting OpenFlow as signaling protocol. In addition, Table 1 reports the deployment, the data and events required by each module to properly work and interact. For example, the attachment detection occurs at Layer-3 by intercepting the Router Solicitation message and sending it to the NC as the payload of an OpenFlow PacketIn message. Such module broadcasts an *MN attachment* event to the other modules which contains the MN's ID and point of attachment. This event is received by the centrally-deployed Binding module which selects the MN's network prefixes and update accordingly the binding cache. Once the binding cache is correctly updated, the module broadcasts a *Binding up* message which includes the MN's prefixes and associated DMM-GWs. Such event is exploited by *i)* the centralized prefix advertisement module to send a Router Advertisement to the MN, and by *ii)* the traffic steering module to update the VLANs tag in the DMM-GWs and ERs.

4. Analytic evaluation

In this section, the PMIPv6 and SDN-based solutions are analyzed considering the signaling overhead and the overall handover latency. We further analyze the scalability of the proposed solutions in terms of size of the forwarding tables which has an impact on the state that network nodes need to keep to properly operate. Table 2 summarizes the notation used throughout this section.

4.1. Signaling cost

The signaling cost is formulated as the overhead in bytes associated to the control messages transmitted when an MN performs a handover. The purpose of this study is to provide a statistical estimation of the signaling rate in B/s, based on mobility and traffic models available in the scientific literature. For this reason, we

Table 2
Notation.

Symbol	Description
MN	Mobile Node
DMM-GW	Distributed Mobility Management Gateway
ER	Egress Router
CMD	Control Mobility Database
NC	Network Controller
RS	An IPv6 Router Solicitation message
RA	An IPv6 Router Advertisement message
RTT	Round Trip Time
G	set of DMM-GWs
$g_j \in G$	element of set G
$G^A \subseteq G$	set of active DMM-GWs at each handover
$g_i^A \in G^A$	element of set G^A
$c(g_i^A)$	signaling cost associated to node g_i^A
C_h	total handover signaling cost
E	set of Egress Routers
$e_m \in E$	element of set E
S_X	entity X's forwarding table size (No. of rules)
U	set of MNs connected to the domain
$U^{g_i \subseteq U}$	set of MNs associated to DMM-GW g_i
$U^{e_m} \subseteq U$	set of MNs associated to ER e_m

apply the analytical framework carried out in [8], which properly captures the peculiarities of DMM protocols. The formal definition of the cost model is expressed in the following. Let G be the set of DMM-GWs deployed in a given area, and $g_i \in G$ its elements. When a mobile node hands off, only few DMM-GWs are involved in the signaling, let $G^A = \{g_1^A, \ldots, g_N^A\}$ be the set of *active DMM-GWs* participating in a handover control sequence. Clearly, $G^A \subseteq G$, and $N = |G^A|$. The set G^A changes at each handover, and its elements g_i^A are reverse-ordered from the latest to the first active DMM-GW visited by the MN. Thence, a_1 is the handoff target DMM-GW and a_2 is the source DMM-GW. It should be noted that, in the PMIPv6-based solution, the size N varies at each handover, whereas for the SDN solution we have $N = 2$ for each handover.

We now characterize the handover cost in terms of signaling load. For any given $g_i^A \in G^A$, we define $c(g_i^A) : G^A \mapsto \mathbb{N}$ as the cost in bytes of each information exchange that involves g_i^A, including the IPv6 and transport-layer headers, but excluding the data link and MAC layer headers.

Therefore, we model the handover cost as follows:

$$C_h = \sum_{g_i^A \in G^A} c(g_i^A), \tag{1}$$

and the function $c(g_i^A)$ is solution-dependent. Its characterization will be addressed in the following paragraphs.

PMIPv6-based

To properly formalize $c(g_i^A)$, we detail some operations from the protocol description (Section 2.1). At each handover, the target DMM-GW transmits a PBU message with the PMIPv6 mandatory options only (denoted as π_{PBU}) to the CMD to notify the MN's new attachment. The CMD replies with a PBA including the mandatory options (π_{PBA}) plus an instance of the *anchor option* ($\pi_{anchor}^{\text{option}}$) for every old DMM-GW that is still anchoring active IP flows. Similarly, the source DMM-GW, and all the other DMM-GWs that are still anchoring IP flows, receive from the CMD a PBU message with the mandatory options plus an instance of the *serving option* ($\pi_{serving}^{\text{option}}$), indicating the new serving DMM-GW. These DMM-GWs then reply to the CMD with a PBA containing the same options to conclude the operation. Therefore we have:

$$c(g_i^A) = \begin{cases} \pi_{PBU} + \pi_{PBA} + (N-1)\pi_{anchor}^{\text{option}} & \text{if } i = 1 \\ \pi_{PBU} + \pi_{serving}^{\text{option}} + \pi_{PBA} + \pi_{serving}^{\text{option}} & \text{if } i \geq 2 \end{cases}. \tag{2}$$

Table 3
Signaling messages cost.

Packet	Bytes	Description
π_{PBU}	128	PBU with mandatory options only
π_{PBA}	128	PBA with mandatory options only
$\pi_{anchor}^{\text{option}}$	56	Previous DMM-GW mobility option
$\pi_{serving}^{\text{option}}$	24	Current DMM-GW mobility option
σ_{RS}	178	Router Solicitation sent to the NC
σ_{RA}	218	Router Advertisement sent by the NC
σ_{write}	264	OpenFlow message for writing a rule
σ_{delete}	232	OpenFlow message for deleting a rule

Thus, Eq. (1) for the PMIPv6-based case turns into:

$$C_h^{\text{PMIPv6-based}} = N(\pi_{PBU} + \pi_{PBA}) + (N-1)\left(\pi_{anchor}^{\text{option}} + 2\pi_{serving}^{\text{option}}\right). \tag{3}$$

In conclusion, this solution's cost depends linearly on the number $N = |G^A|$ of active DMM-GWs.

The value of N depends on both the MN mobility (i.e., handover frequency) and traffic patterns. It is intuitive that the more often the MN changes attachment point, the larger is the number of active DMM-GWs. However, a DMM-GW is eventually de-activated when there are no more MN's IP flows traversing it. So, the longer the IP flows started by the MN are, the longer is the DMM-GW's activity interval. Knowing the statistical distribution of the handover rate and how long an IP flow is maintained by the DMM-GW anchoring that flow permits to compute the statistical distribution of the number of active DMM-GWs at any time [8], and thus the size of the set G^A. In this paper we simplify the problem assuming that an MN spends an exponential time with mean value μ attached to a DMM-GW before handing over to a different one. Besides, we assume that after a handover, an old DMM-GW remains active for an exponential interval of mean λ. Using the results reported in [8], we obtain $\overline{N} = \mathrm{E}[N]$, as:

$$\overline{N} = 2 + \frac{\lambda}{\mu}. \tag{4}$$

SDN-based

In the SDN-based solution, the only DMM-GWs involved during a handover are the source and target DMM-GWs, thus $G^A = \{G_1^A, G_2^A\}$ for every handover. From the protocol description in Section 3, upon a handover, the NC writes on the target DMM-GW two downlink rules and as many uplink rules as the number of active ERs for the mobile node. Moreover, the NC writes two downlink rules on each ER, and removes the uplink and downlink rules on the source DMM-GW. The reason of having two downlink rules is given by the need of properly identifying the mobile node's IPv6 local and global addresses.[6] Thus, for the general case of having k ERs, and using the message notation shown in Table 3, the cost function is defined as follows:

$$c(g_i^A) = \begin{cases} \sigma_{RS} + \sigma_{RA} + (3k+2)\sigma_{write} & \text{if } i = 1 \\ (k+2)\sigma_{delete} & \text{if } i = 2 \end{cases}. \tag{5}$$

Therefore, Eq. (1) for the SDN-based case turns out to be:

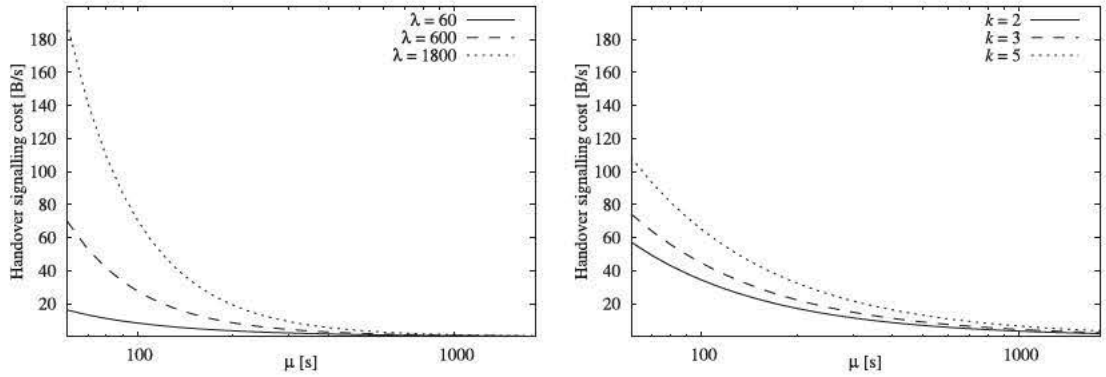
$$C_h^{\text{SDN-based}} = \sigma_{RS} + \sigma_{RA} + (3k+2)\sigma_{write} + (k+2)\sigma_{delete}, \tag{6}$$

from which it can be observed that the SDN solution's cost depends linearly on k.

Signaling cost considerations

We analyze next the average signaling cost for a single MN, for the two solutions. We consider the average residence time μ as

[6] IPv6 uses unicast and multicast addresses to reach the mobile node. Even if the identification would be based on MAC addresses, it would still require two rules.



(a) PMIPv6-based solution (b) SDN-based solution

Fig. 3. Handover signaling cost for the two DMM solutions.

defined previously for the PMIPv6-based solution, and C_h/μ as the solution's cost in bytes per second. The size of each message involved in Eq. (2), Eq. (5), has been measured experimentally (and its value is reported in Table 3). The description of the experiments is reported later in Section 5.

The signaling cost is reported, for different values of λ and k, in Fig. 3(a) for the PMIPv6-based solution and in Fig. 3(b) for the SDN-based. We observe a performance degradation on the PMIPv6-based solution for large values of the ratio λ/μ values, and hence large $\overline{N}$. This is a scenario with high mobility and long lived IP flows. In order to cope with this limitation, the deployment of such solution should jointly consider the coverage area and the level of mobility of MNs. That is, the DMM-GW's coverage area should not be too small in order to reduce the number of active DMM-GWs. Such information is usually available to operators. Nevertheless, this solution is more suitable when handling scenarios with low mobility, i.e., for high values of μ, or short lived flows, i.e., for low λ.

The SDN-based solution behaves in a more predictable way, as it only depends on the value k and it is independent of the traffic pattern of MNs. Operators have the profiles of each MN, therefore the value k can be also adapted on an MN basis. As a result, the network can be managed in a smarter way and a higher network's efficiency can be achieved by spreading the MNs on multiple ERs.

4.2. Forwarding table size

We characterize next the parameter S as the size, i.e., the number of rules, of the forwarding table of the involved network nodes. The number of forwarding entries represents the state that each node needs to keep in the network to properly forward the traffic to the MNs. At this end, we define U as the set of subscribers in the domain, and, $U^{g_i} \subseteq U$ the set of MNs connected to a generic DMM-GW g_i. We assume that users are uniformly distributed among DMM-GWs, so that $|U^{g_i}| = |U|/|G|$.

PMIPv6-based

As described in Section 4.1 and according with the statistical framework defined in [8], an MN has on average $\overline{N}-1$ act an MN has $\overline{N}-1$ active prefixes advertised by old DMM-GWs,for which 3 routing rules are necessary: one at the anchor DMM-GW for downlink forwarding, and two at the current DMM-GW, respectively for uplink and downlink. In addition, the MN configures an IPv6 prefix from the current DMM-GW's pool, implying one downlink routing rule at the current DMM-GW. By dividing the total number of routing rules for the number of DMM-GWs, we obtain that, on average, the number of routing entries in a DMM-GW is given by:

$$S_{g_i}^{\text{PMIPv6-based}} = \left(3\overline{N}-2\right)|U^{g_i}|\,. \tag{7}$$

SDN-based

For the SDN-based solution, the study applies to the DMM-GWs and ERs, as they are the only involved nodes. Let E be the set of egress routers deployed in a domain and $e_m \in E$ its elements. While a DMM-GW manages no more than the U^{g_i} MNs directly attached to it, an ER manages the traffic of MNs that might be connected to multiple DMM-GWs. As described in Section 4.1, on each ER the NC writes two OpenFlow rules for each MN. Therefore, the forwarding table's size on the m-th ER, e_m, is independent of k and depends only on the number of MNs managed by that ER. We denoted this set as $U^{e_m} \subseteq U$. As a result, the size of the forwarding table turns into:

$$S_{e_m}^{\text{SDN-based}} = 2|U^{e_m}|\,. \tag{8}$$

It is worth highlighting that this is the best achievable result. In fact, the smallest number of rules necessary to properly identify a single MN is two rules as explained in the previous section. On the contrary, the forwarding table on a DMM-GW depends on k and on U^{g_i}. Indeed, the NC writes $k+2$ rules on the target DMM-GW for each associated MN, leading to:

$$S_{g_i}^{\text{SDN-based}} = (k+2)|U_{g_i}|\,. \tag{9}$$

Regarding U^{e_m} and U^{g_i}, we can safely assume:

$$|U| \geq |U^{e_m}| \gg |U^{g_i}| \qquad \forall i, m\,, \tag{10}$$

that is the number of MNs managed by an ER is much larger than the number of MNs managed by a single DMM-GW (up to all the MNs in the domain). Therefore, k does not present a major scalability problem for the DMM-GW's forwarding table size.

4.3. Handover latency

Next, we study how the protocol operations affect the handover latency for each of the two studied DMM solutions.

The handover delay analysis can be split into three subproblems: *i)* the Layer-2 handover, including the time elapsed since the old radio link is torn down until the new one is established, *ii)* the Layer-3 configuration, considering the time required by the MN to obtain network layer connectivity (including the Layer-2 handover), and *iii)* the IP flow recovery, i.e., the interval during which an IP flow is interrupted due to the handover (including both the Layer-2, the Layer-3 configuration, plus then the remaining actions performed within the network to ensure IP session continuity).

In all our protocols, the Layer-2 handover does not depend on the specific solution and it is the same for all of them, thus we omit it in the equations. Nevertheless, in Section 5.2 we provide the results obtained in our experiments.

PMIPv6-based

The MN establishes the Layer-3 connectivity by requesting an IPv6 prefix with a Router Solicitation (RS) message. The DMM-GW, before sending to the mobile node the IPv6 prefix information in a Router Acknowledgement (RA), performs a two-way message exchange with the CMD to register the MN presence and the assigned prefix. The message from the CMD contains the necessary parameters to set up the tunnels and the routing towards the old DMM-GWs that are anchoring the MN's active prefixes. As a result, the time required by the MN for the Layer-3 configuration is due to the round trip time (RTT) between the MN and the DMM-GW for the RS/RA exchange, plus the RTT between the DMM-GW and the CMD for PBU/PBA signaling and, finally, a processing time $T_P^{\text{PMIPv6-based}}$ for each of the N active DMM-GWs. This Layer-3 latency can then be expressed as:

$$T_{L3}^{\text{PMIPv6-based}} = RTT_{MN-DMMGW} + RTT_{DMMGW-CMD} + N\,T_P^{\text{PMIPv6-based}}\,. \tag{11}$$

In order to recover the IP flows started with the IPv6 prefixes assigned by previous DMM-GWs, the CMD instructs all the previous active DMM-GWs with parallel PBU/PBA signaling after the attachment notification from the new DMM-GW is received. For the model, we can then assume that $RTT_{DMMGW-CMD}$ is constant for all the DMM-GWs, so that the new DMM-GW and all the old ones receive the update message from the CMD simultaneously. Next, an old DMM-GW re-builds the data path with a tunnel to the current MN's DMM-GW in a time $T_P^{\text{PMIPv6-based}}$, and then packets flow to the serving DMM-GW in a time $(1/2)RTT_{DMMGW-DMMGW}$. We then obtain:

$$T_{flow-recovery}^{\text{PMIPv6-based}} = T_{L2-ho} + RTT_{MN-DMMGW} + RTT_{DMMGW-CMD} + T_P^{\text{PMIPv6-based}} + \frac{1}{2}RTT_{DMMGW-DMMGW}\,. \tag{12}$$

SDN-based

In this case, the Router Solicitation (RS) sent by the mobile node upon attachment is intercepted by the target DMM-GW and forwarded to the NC. At this point, the Network Controller (NC) configures the forwarding path in the network by: *i)* writing the rules on the new DMM-GW and on the Egress Routers (ERs), and, *ii)* deleting the rules on the old DMM-GW. In this solution, the order plays an important role, indeed, after the configuration of the target DMM-GW and of the ERs, the path in the network is updated and the traffic is finally able to reach the MN. Consequently, to compute the IP flow-recovery time, we can get rid of the time necessary to delete the rules on the last visited DMM-GW. After the configuration phase, the NC generates an RA which is sent back to the DMM-GW and finally forwarded by the latter to the MN. This RS/RA exchange lasts an RTT between the MN and the DMM-GW plus another RTT between the DMM-GW and the NC. In the new DMM-GW, the NC writes $k+2$ rules through $k+2$ parallel messages, where k is the number of ERs. As a result, the NC takes $(1/2)RTT_{DMMGW-NC}$ to configure the new DMM-GW. For the ERs, the NC writes two rules on each ER in parallel, taking $(1/2)RTT_{ER-NC}$ to update the rules, where RTT_{ER-NC} is the distance between the ER and the NC. For simplicity, we consider RTT_{ER-NC} as constant for each ER. Albeit the NC configures the rules in parallel, the generation of those messages is performed sequentially. We denote as $T_P^{\text{SDN-based}}$ the processing time required by the NC to forge the OpenFlow messages for each of the k ERs. Therefore, the Layer-3 configuration latency is:

$$T_{L3}^{\text{SDN-based}} = RTT_{MN-DMMGW} + \frac{3}{2}RTT_{DMMGW-NC} + \frac{1}{2}RTT_{ER-NC} + k\,T_P^{\text{SDN-based}}\,. \tag{13}$$

After the configuration phase and the reception of the Router Acknowledgement message by the MN, the packets can finally reach the MN, taking a time $T_{transport}$. Hence, the flow-recovery time is:

$$T_{flow-recovery}^{\text{SDN-based}} = T_{L2-ho} + T_{L3}^{\text{SDN-based}} + T_{transport}\,. \tag{14}$$

5. Experimental evaluation

Complementing the analysis conducted before, in this section we describe the experimental evaluation of the PMIPv6-based and the SDN-based DMM solution, aiming at providing a proof of concept to assess the solutions' feasibility and performance.

5.1. Test-bed description

In order to evaluate the two solutions, we deployed a test-bed for each, based on GNU/Linux machines connected through an Ethernet network. Each test-bed comprises a set DMM-GWs providing IEEE 802.11b/g wireless access to the MNs. The systems are tested for three different configurations, employing 2, 3 or 5 DMM-GWs. As none of our solutions devises any intervention on the MN, the hardware and software requirements for the MNs are loose, being simply an IEEE 802.11b/g wireless card, and a standard IPv6 stack implementing Neighbor Discovery [14]. In the following paragraphs we delve into the solution-specific test-beds description.

5.1.1. PMIPv6-based test-bed

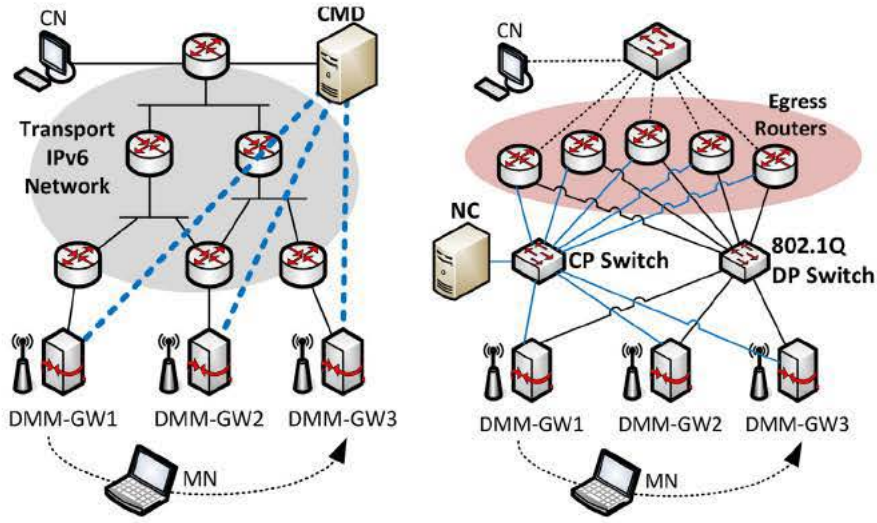
The PMIPv6-based test-bed is depicted in Fig. 4(a). In order to make the scenario more realistic, we added to the test-bed a transport IPv6 network composed by several IPv6 routers. These routers connect the DMM-GWs to the CMD and to the CN. The dashed blue lines in Fig. 4(a) represent the logical interaction between the CMD and the DMM-GWs (these lines do not represent a dedicated path between the CMD and the DMM-GWs). The DMM-GWs and the CMD are the only nodes that run our implementation of the PMIPv6-based solution. Such implementation replicates the signaling and operations specified in [7] and briefly summarized in Section 2.1.

5.1.2. SDN-based test-bed

In the SDN-based test-bed, in addition to the DMM-GWs, we added 5 Egress Routers (ERs) as illustrated in Fig. 4(b). As can be observed from the picture, the DMM-GWs and the ERs are connected each other through two separate networks, one for the control plane (drawn with blue lines) and one for the data plane (the black solid lines). In the control plane network, a switch realizes the interconnection among all the nodes and also with the NC (see the CP Switch node in Fig. 4(b)). For the data plane, the packet forwarding within the network is based on VLANs and statically configured. Thus, we deployed and configured an 802.1Q-capable switch in the data plane that interconnects all the DMM-GWs and ERs. Moreover, the ERs have a third link used to connect the test-bed to the CN.

Since the SDN-based solution uses OpenFlow as Southbound API, all the DMM-GWs and ERs run the version 3.10 of Linux kernel. This version of the kernel includes Open vSwitch[7] which provides an OpenFlow 1.3 interface. The NC runs Ryu[8] as OpenFlow controller. The SDN-based solution is therefore implemented as Ryu application (i.e., based on the API provided by the NC). The connection between Open vSwitch and Ryu is performed out-of-band involving TCP for the OpenFlow messages delivery. The application is in charge of all the tasks described in Section 3.

[7] http://openvswitch.org/.
[8] http://osrg.github.io/ryu/.



(a) PMIPv6-based test-bed (b) SDN-based test-bed

Fig. 4. Test-beds used in the experimental evaluation.

Table 4
Handover latency experimental results in milliseconds.

Type of solution		Layer-2 ho.		Layer-3 conf.		IP flow rec.	
		Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev
PMIPv6-based	$N = 2$	12.9	4.4	25.8	8.6	40.3	9.3
	$N = 3$	12.9	4.4	27.8	8.7	42.5	9.4
	$N = 5$	12.9	4.4	32.6	8.9	47.4	9.7
SDN-based	$k = 2$	12.9	4.4	26.8	2.3	33.0	2.8
	$k = 3$	12.9	4.4	29.2	2.4	35.6	3.0
	$k = 5$	12.9	4.4	33.7	4.4	40.2	4.8

5.2. Experimental results

For both implementations, we have measured the three handover events introduced in Section 4.3. We have used Wireshark[9] as packet sniffer, installed in the MN to measure the intervals detailed in the following:

1. *Layer-2 handover.* It is measured as the interval between two IEEE 802.11 control messages: *Deauthentication*, sent by the MN to the old DMM-GW, and *Association response* received by the MN from the new DMM-GW.
2. *Layer-3 configuration.* It is the time spent since the *Deauthentication* message, to the instant when an RA message is received by the MN[10].
3. *IP flow recovery.* It is measured as the time required to recover *ping* traffic generated by a correspondent node to the MN every 2 ms, which is below the average the MN-CN RTT. This corresponds to the interval between the last *ping* packet received or sent by the MN before the handover and the first *ping* packet received or sent after the handover.

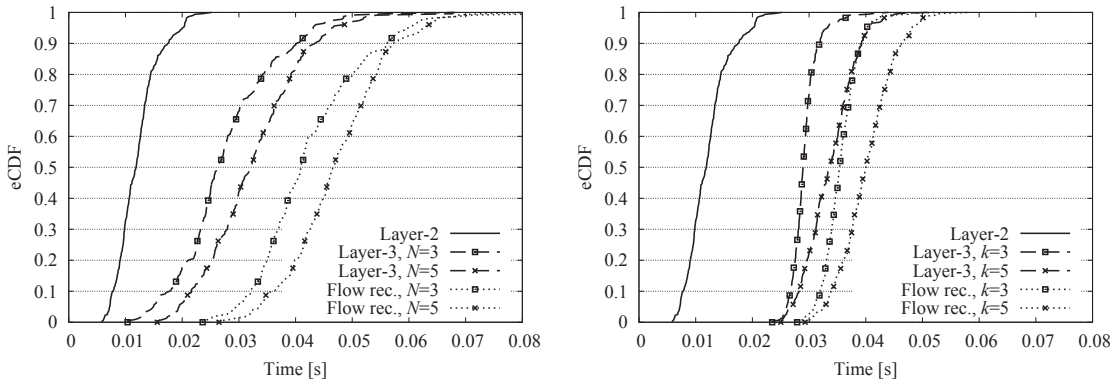
Fig. 5 depicts the empirical CDF of the above components from the values obtained from few hundreds handovers when $N = 3, 5$ in the PMIPv6-based case, and $k = 3, 5$ in the SDN-based case. Table 4 summarizes the experimental results reporting the mean and the standard deviation values also in the cases $N = 2$ and $k = 2$.

As expected, the *Layer-2 handover* does not depend on the mobility protocol. The table reports the same value for all setups because the measured difference was negligible in our testing system. For what concerns the *Layer-3 configuration*, i.e., how long it takes for the MN to gain IP connectivity with the DMM-GW, it can be observed that the two protocols behave similarly on average, showing a lower variance in the SDN case due to some system-level optimizations applied to the node acting as network controller. The *IP flow recovery* (i.e., *ping*) time exhibits the largest gain in favor of the SDN approach. This is due to the use of tunneling by the PMIPv6-based solution, which was observed to introduce, on average, around 15 ms of additional delay with respect to the Layer-3 configuration, for all values of N. On the contrary, the SDN-based solution accomplishes the ping recovery with less than 7 ms of additional delay.

In order to better understand how the two solutions scale, Fig. 6 explores in detail the components of the Layer-3 configuration time for varying values of the number of active DMM-GWs, $N = 2, 3, 5$ in the PMIPv6-based case, and the number of egress routers, $k = 2, 3, 5$ in the SDN-based case. As it can be noticed from the results, the Layer-2 switch time is the major contributing term in all setups. More, we observed a 5 ms gap, denoted as "MN gap", between the instant the MN receives the *Association response* message, and the time it sends the RS message to the DMM-GW. This gap could be removed by employing a dedicated detection mechanism for the Layer-2 link activation and de-activation. After the link-up phase, we could separate the component due to message transmission, which depends on the sum of the RTT in the radio link between the MN and the DMM-GW, and the RTT in the wire between the DMM-GW and the CMD or NC, respectively for the

[9] Wireshark, http://www.wireshark.org/.

[10] The IPv6 Duplicate Address Detection is disabled since the prefix is uniquely assigned to the MN, thus it is not a necessary process.



(a) PMIPv6-based solution (b) SDN-based solution

Fig. 5. Flow-recovery time eCDF.

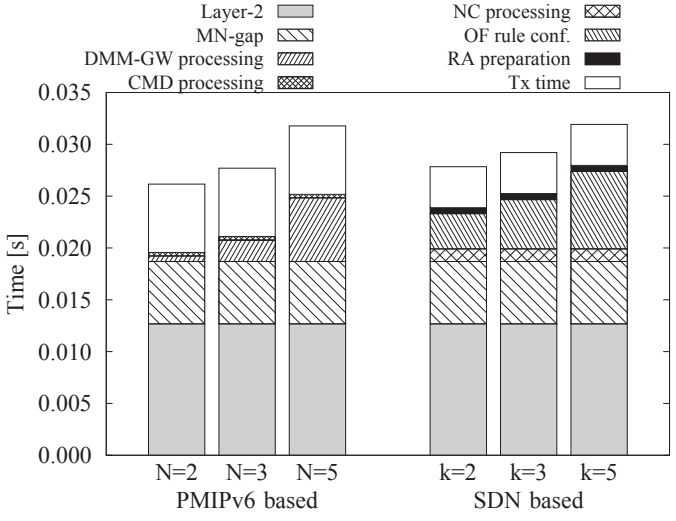


Fig. 6. Layer-3 latency composition.

PMIPv6-based or the SDN-based solution. In our laboratory tests, all the nodes are close to each other, and such RTT sum is less than 5ms. In a real deployment, with larger RTT values, the Layer-3 configuration time would tend to approximate the air time plus the distance from the central node to the farthest router involved in the signaling. The above components do not significantly vary with the increasing number of N and k, whereas such parameters impact the processing at the network nodes, confirming the intuition that T_P tends to grow with the number of entities involved in the handover operations. In the PMIPv6-based case, the heaviest burden is on the DMM-GW, because of the tunnels and routes set up, so, the larger is the number of previous DMM-GWs, the longer is the latency. The CMD is mainly answering to a query, so its task is accomplished much quicker in approximately constant time. In the SDN based case, the NC has to compute and send the rules to the ERs. In addition it has to process the RS from the MN and prepare the RA message. From Fig. 6, it can be observed that the variable components approximately grow linearly in both solutions.

The results reported in this section show that the SDN-based DMM approach has similar or better performance figures than previous DMM protocols based on PMIPv6. This is an encouraging outcome to foster further optimizations for future deployments of such kind of solutions. For example, further evaluation is required to analyze scenarios where multiple MNs simultaneously attach to the network or perform a handover, thus producing various requests overlapping in time at the NC side. For instance, in these cases the NC is expected to be backlogged because of all the concurrent MN requests. Similarly, multiple flow rules are expected to be configured at the same time on the same SDN switch, e.g., when several MNs simultaneously perform an handover to the same target DMM-GW. These would eventually lead to an increase of processing and configuration time, thence to a higher overall handover latency experienced by the MNs. Evidently, a SDN-based DMM solution should consider also the distribution of the control plane in addition to the data plane. This implies having multiple replicas of the DMM solution modules in the network which still need to provide a harmonized mobility support to the MNs whilst providing a bound to the handover latency. Therefore, an analogous methodology as the one used in this section can be then used in such scenarios to analyze the breakdown of the handover latency and to derive some deployment options for the SDN-based DMM solution, e.g. the number of replicas and where to deploy them to support a higher volume of MNs and different mobility patterns. These topics are left for future work by the authors.

6. Related work

SDN has raised in popularity very recently, and it has attracted the attention of the research community, as it eases testing protocols and networking algorithms. The flexibility and programmability of SDN architectures have contributed to the proliferation of several large deployments designed by leading research institutions. Among the first deployments we find B4 [24], which is Google's SDN-based wide area network (WAN) to interconnect its data centers around the world. For B4, an extensive analysis is available on how to manage the routing and traffic engineering through OpenFlow and the designers of B4 provide interesting insights on design, performance, scalability and failure resilience of their solution. Although providing mobility is not within the objectives of B4, we have leveraged the knowledge provided by the B4 large scale implementation example to design our SDN solution.

Despite the usage of a wired structure to transfer the OpenFlow signaling in our network, we focus on the wireless access and the mobility management. In that regard, one of the most remarkable SDN deployments applied to wireless networking is *OpenRoads* [25] (also known as *OpenFlow Wireless*) developed at Stanford University, open to researchers for running their algorithms concurrently by means of virtualization. OpenRoads incorporates different wireless technologies, namely WiFi and WiMAX, and one of its early proofs of concept was based on providing mobility across multiple technologies [26]. In addition, the performance of

OpenRoads has been demonstrated by means of an n-casting transmission solution [27]. All these approaches are based on the same principle as our mobility approach, reconfiguring the data-path, although they do not consider IP mobility or the design of a scalable architecture as we do. All the tools used by OpenRoads are open source, so as to make the infrastructure reproducible by other research groups in their own networks. Likewise, our implementation shows the flexibility of current SDN software tools available as open source and is built upon commercial-off-the-shelf devices. At the moment we only focus on IEEE 802.11 access points, but we are planning to extend our test-bed to include heterogeneous access technologies in the short term.

A full software-defined mobile network (SDMN) is defined in MobileFlow [28], and its authors provide a comparison to the current Evolved Packet Core (EPC) architecture. Although no numeric results are reported, a prototype implementation is also proposed in [28]. The purpose is to show a proof of concept of the MobileFlow Forwarding Engine (MFFE), which encompasses all the user plane protocols and functions, and the MobileFlow Controller (MFC), which is a logically centralized entity that configures dynamically the MFFEs (i.e., the data plane). Despite MobileFlow is OpenFlow-based, MFFEs must also support operations that are not carried out at the switch-level, as layer-3 tunneling, for instance. Mobility management can be supported as the controller can update forwarding rules according to the tunnel encapsulation or decapsulation requirements. This approach is also followed in our implementation, where we can set the tunneling and forwarding rules above link layer and the controller updates the forwarding rules in the OpenFlow domain. A different approach presented in [29] proposes to move the EPC to the cloud by means of virtualization and implementing GTP extensions for OpenFlow for mobility management. The mobility solutions proposed by these works are not really inspired by the DMM paradigm, but are rather based on the same mobility concepts currently used in cellular networks, hence inheriting the scalability issues of traditional mobility approaches such as PMIP or GTP-based mobility management.

Regarding mobility management solutions relaying on the SDN paradigm for session continuity management, in [30] the authors propose a solution based on IP translation across the mobility domain. This solution differs mainly with ours on the mobile terminal support. While our solution is completely transparent for the mobile terminal, the solution presented in [30] requires the terminal to bind its current location to its identifier, using Mobile IP signaling. The solution is supported by a mininet[11]-based proof of concept, which authors use to observe the behavior of TCP flows during handovers handled by their solution, compared to plain PMIPv6.

A similar approach can also be found in [31]. This work uses a SDN-based approach for path modification, but it also provides of extensions to OpenFlow to update the DNS of the network with the new location of the user. The use of this extensions highly impact on the handover performance which is heavily increased with respect to the solution proposed in the present paper.

Similar approaches as the one defined in the paper above can also be found in the literature. For example, authors in [32] propose a system using OpenFlow to manage the mobility of users in 3GPP networks. Differing from our SDN proposal, this work is a conceptual analysis on how SDN can be applied to 3GPP networks and does not include any implementation or empirical evaluation. Nevertheless, the same authors further elaborated their solution, leading to [33]. The paper proposes an SDN-based DMM approach for virtualized LTE systems which leverages on an interface between the SDN controller and the MME, in order to detect an IP anchor change (i.e., a PGW relocation). When such a relocation occurs, the SDN controller re-routes ongoing traffic to the new PGW. The evaluation focuses on the handover latency produced by this solution, using NS-3 simulations whose results are aligned with the ones experimentally obtained in the present paper. Similar to the experimental validation conducted in the present paper, another testbed-based validation using commodity hardware and WiFi access is available in [34]. The solution therein proposes a hierarchy of SDN controllers in order to handle intra-district handovers (i.e., within an access network handled by the same DMM-GW) and inter-district handovers (i.e., including DMM-GW relocation). Due to the scenario discussed by the solution and additional complexity required in the signaling, the results reported in [34] for the handover latency appear slightly larger than those presented in the present research. Yet another SDN-based DMM architecture is validated experimentally through a setup employing a Ryu controller and a mininet-created test network in [35]. The SDN solution therein handles packet redirection after handover in two different ways, both different from our proposal. The first method is the tunnel mode, i.e., establishing a tunnel between the source and target access gateways in order to convey re-directed packets; the second method is based on route optimization, that is a full path computation and the subsequent population of the forwarding rules onto the in-path switches. The numeric results show similar values for the handover latency as those obtained in the present paper, but they are not directly comparable as there is no wireless access employed in the testbed described in [35]. In addition, we argue that the redirection methods do not scale as well as the one proposed in the present solution, especially the method employing a full path reconfiguration.

7. Conclusion

Distributed Mobility Management is seen as a necessary tool to design future mobile network deployments, in order to offload the network core from traffic that can be locally routed close to the access. Due to the foreseen increase in the access capacity in future networks, reducing the amount of traffic traversing the core of the network is of the upmost importance to avoid a capacity crunch at the operator infrastructure. Different actors have been working on this area, being the IETF a major venue where most of the solutions have been discussed so far, while 3GPP and ONF have more recently started to work on distributed mobility architectures. Although there have been many different proposals, most of them share a characteristic: they are an evolved version of current IP mobility based solutions. While these are enough to offload the network core, and pose no significant deployment concerns, operators are already looking into the next stage: *software networks*. The SDN paradigm has gained a lot of attention from operators, as it can reduce the complexity and costs incurred by service creation and network operation. Therefore, it is important to understand how an SDN-based solution might look like when providing DMM support.

The main contribution of this paper is the analytic and experimental evaluation of two key DMM protocol families: IP mobility and SDN based, by designing, modeling and implementing a particular solution belonging to each of the identified protocol categories. Additionally, we walk the path of decomposing the functions that a DMM solution should have and identify how these can be implemented in an SDN-based solution. The two DMM protocol families are analyzed by using as representative solutions two approaches designed and implemented by the authors of this paper. Existing state-of-the-art solutions are not generally studied both analytically and experimentally, so we believe this paper provides solid insights on how to apply DMM concepts in future mobile networks.

[11] http://mininet.org.

The results obtained from analysis and experiments show that the performance of the analyzed solutions depends on the scenario being considered, but also indicate that SDN approaches have a big potential: *i)* achievable performance is good and even better than the one of the PMIPv6-based solution; *ii)* the solution can be easily implemented, and; *iii)* provides additional flexibility in regards of how it behaves and provides service differentiation.

Acknowledgment

This work has been funded by the European Union's Horizon 2020 programme under the grant agreement no. 671598 "5G-Crosshaul: the 5G integrated fronthaul/backhaul".

References

[1] CISCO, Cisco Visual Networking Index: Forecast and Methodology, 2015–2020, 2016. White Paper
[2] NGMN Alliance, 5G White Paper, Technical Report, Next Generation Mobile Networks (NGMN) Alliance, 2015.
[3] 3GPP, Service Requirements for Next Generation New Services and Markets, TS 22.261, 3rd Generation Partnership Project (3GPP), 2016.
[4] AT&T, ECOMP (Enhanced Control, Orchestration, Management & Policy) Architecture White Paper, Technical Report, AT&T Inc., 2016.
[5] 3GPP, Study on Architecture for Next Generation System, TR 23.799, 3rd Generation Partnership Project (3GPP), 2016.
[6] H. Chan, D. Liu, P. Seite, H. Yokota, J. Korhonen, Requirements for Distributed Mobility Management, 2014, (RFC 7333).
[7] C.J. Bernardos, et al., A PMIPv6-based solution for Distributed Mobility Management, 2014, IETF draft, draft-bernardos-dmm-pmip-03.
[8] F. Giust, C. Bernardos, A. de la Oliva, Analytic evaluation and experimental validation of a network-based IPv6 distributed mobility management solution, IEEE Trans. Mobile Comput. 13 (11) (2014) 2484–2497, doi:10.1109/TMC.2014.2307304.
[9] C. Perkins, D. Johnson, J. Arkko, Mobility Support in IPv6, 2011, (RFC 6275).
[10] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, B. Patil, Proxy Mobile IPv6, 2008, (RFC 5213).
[11] P. Seite, et al., Dynamic Mobility Anchoring, 2014, IETF draft, draft-seite-dmm-dma.
[12] H. Chan, et al., Enhanced mobility anchoring, 2014, IETF draft, draft-chan-dmm-enhanced-mobility-anchoring.
[13] J.H. Lee, J.M. Bonnin, X. Lagrange, Host-based distributed mobility management support protocol for ipv6 mobile networks, in: 2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2012, pp. 61–68, doi:10.1109/WiMOB.2012.6379140.
[14] T. Narten, E. Nordmark, W. Simpson, H. Soliman, Neighbor Discovery for IP version 6 (IPv6), 2007, (RFC 4861).
[15] ONF, OpenFlow Switch Specification: Version 1.5.1, TS 025, Open Networking Foundation, 2015.
[16] ONF, SDN Architecture, TR 502, Open Networking Foundation, 2015.
[17] 3GPP, S1 Application Protocol (S1AP), TS 36.413, 3rd Generation Partnership Project (3GPP), 2015.
[18] F.e. a. Droms, Dynamic Host Configuration Protocol for IPv6 (DHCPv6), 2003, (RFC 3315).
[19] 3GPP, General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U), TS 29.281, 3rd Generation Partnership Project (3GPP), 2016.
[20] IEEE, Virtual Bridged Local Area Networks, IEEE Standards for Local and metropolitan area networks 802.1Q, Institute of Electrical and Electronics Engineers (IEEE), 2003.
[21] E. Rosen, A. Viswanathan, R. Callon, Multiprotocol Label Switching Architecture, 2001, (RFC 3031).
[22] 3GPP, Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C), TS 29.274, 3rd Generation Partnership Project (3GPP), 2011.
[23] C. Perkins, IP Mobility Support for IPv4, 2002, IETF RFC 3344.
[24] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, A. Vahdat, B4: Experience with a Globally-deployed Software Defined WAN, in: SIGCOMM '13, ACM, New York, NY, USA, 2013, pp. 3–14, doi:10.1145/2486001.2486019.
[25] K.-K. Yap, M. Kobayashi, D. Underhill, S. Seetharaman, P. Kazemian, N. McKeown, The Stanford OpenRoads Deployment, in: WINTECH '09, ACM, New York, NY, USA, 2009, pp. 59–66, doi:10.1145/1614293.1614304.
[26] K.-K. Yap, R. Sherwood, M. Kobayashi, T.-Y. Huang, M. Chan, N. Handigol, N. McKeown, G. Parulkar, Blueprint for Introducing Innovation into Wireless Mobile Networks, in: ACM SIGCOMM VISA Workshop 2010, ACM, New York, NY, USA, 2010a, pp. 25–32, doi:10.1145/1851399.1851404.
[27] K.-K. Yap, M. Kobayashi, R. Sherwood, T.-Y. Huang, M. Chan, N. Handigol, N. McKeown, Openroads: empowering research in mobile networks, SIGCOMM Comput. Commun. Rev. 40 (1) (2010b) 125–126, doi:10.1145/1672308.1672331.
[28] K. Pentikousis, Y. Wang, W. Hu, Mobileflow: toward software-defined mobile networks, IEEE Commun. Mag. 51 (7) (2013) 44–53, doi:10.1109/MCOM.2013.6553677.
[29] J. Kempf, B. Johansson, S. Pettersson, H. Luning, T. Nilsson, Moving the mobile Evolved Packet Core to the cloud, in: WiMob 2012, IEEE, 2012, pp. 784–791, doi:10.1109/WiMOB.2012.6379165.
[30] Y. Wang, J. Bi, A solution for IP mobility support in software defined networks, in: Computer Communication and Networks (ICCCN), 2014 23rd International Conference on, IEEE, 2014, pp. 1–8.
[31] Y. Li, H. Wang, M. Liu, B. Zhang, H. Mao, Software defined networking for distributed mobility management, in: Globecom Workshops (GC Wkshps), 2013 IEEE, IEEE, 2013, pp. 885–889.
[32] M. Karimzadeh, L. Valtulina, G. Karagiannis, Applying SDN/OpenFlow in Virtualized LTE to support Distributed Mobility Management (DMM), in: 4th International Conference on Cloud Computing and Services Science, CLOSER 2014, SciTePress, 2014, p. 86.
[33] L. Valtulina, M. Karimzadeh, G. Karagiannis, G. Heijenk, A. Pras, Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems, in: 2014 IEEE Globecom Workshops (GC Wkshps), 2014, pp. 18–23, doi:10.1109/GLOCOMW.2014.7063379.
[34] M.I. Sanchez, A. De la Oliva, V. Mancuso, Experimental evaluation of an SDN-based distributed mobility management solution, in: Proceedings of the Workshop on Mobility in the Evolving Internet Architecture, ACM, 2016, pp. 31–36.
[35] T.-T. Nguyen, C. Bonnet, J. Harri, Sdn-based distributed mobility management for 5g networks, in: Wireless Communications and Networking Conference (WCNC), 2016 IEEE, IEEE, 2016, pp. 1–7.

Luca Cominardi received his Bachelor's and Master's degree in Computer Science at University of Brescia, Italy. He did an internship and undertook a Master in Telematics Engineering at University Carlos III of Madrid (UC3M), Spain. Currently he is working at IMDEA Networks Institute and pursuing his Ph.D. at UC3M. His main research interests are Software Defined Networking (SDN), Network Function Virtualization (NFV) and integration of the wireless medium into the two former.

Fabio Giust received a MSc degree in Telecommunications Engineering at University of Padova, Italy and a PhD in Telematics Engineering at University Carlos III of Madrid, Spain. Currently he is working at NEC Europe Ltd., in the German research laboratories. His research interests cover 5G mobile networks, mobile edge computing and network virtualization. He has published several papers in international conferences and journals on IP-based mobility and wireless networks. He is also a contributor to IETF and ETSI.

CARLOS J. BERNARDOS received a Telecommunication Engineering degree in 2003, and a PhD in Telematics in 2006, both from the University Carlos III of Madrid (UC3M), where he worked as a research and teaching assistant from 2003 to 2008 and, since then, has worked as an Associate Professor. His Ph.D. thesis focused on route optimization for mobile networks in IPv6 heterogeneous environments. His current work focuses on vehicular networks and IP-based mobile communication protocols. He has published over 50 scientific papers in prestigious international journals and conferences, and he is also an active contributor to the Internet Engineering Task Force (IETF). He served as TPC chair of WEEDEV 2009 and as TPC co-chair of the Mobility track of NTMS 2011. He has also served as guest editor of IEEE Network.

Antonio de la Oliva obtained the Degree on Telecommunication Engineering by the University Carlos III of Madrid in December 2004. After a 6 month internship on the NEC Network Labs, Antonio de la Oliva started developing its main line of research, mobility on heterogeneous networks, focusing by this time on the analysis and development of the upcoming IEEE 802.21 specification. In July 2008, Antonio de la Oliva presented his PhD. thesis, which obtained, in June 2009, the Alcatel - Lucent award ex-aequo, to the best contribution to new IPTV services, granted by the Royal Telecommunication Engineering Institute of Spain. In the recent years, in addition to participating in several European research projects and serving as principal researcher of a national project, Antonio de la Oliva has served as Vice-chair of the IEEE 802.21b task group and Technical Editor of IEEE 802.21d, contributing significantly to the development of the IEEE 802 standards for Media Independent Handover Services. He is also serving as Conference Organizer of the 2013 IEEE Online Conference on Green Communications (IEEE OnlineGreenComm). Currently, Antonio de la Oliva works as Visiting Professor at the Telematics Engineering department of the University Carlos III of Madrid, where he is performing lecturing and research activities.