Towards Elastic Distributed SDN/NFV Controller for 5G Mobile Cloud Management Systems

Abdelkader Aissioui¹, Adlen Ksentini², Abdelhak Gueroui¹, and Tarik Taleb³

¹PriSM/University of Versailles, France
²IRISA/University of Rennes, France
³Communications and Networking Department/Aalto University, Finland

E-mail: aissiouikader@gmail.com, adlen.ksentini@irisa.fr, mogue@prism.uvsq.fr, talebtarik@ieee.org

Abstract—Future 5G mobile network architecture is expected to offer capacities to accommodate the inexorable rise in mobile data traffic and to meet further stringent latency and reliability requirements to support diverse high data rate applications and services. Mobile Cloud Computing (MCC) in 5G has emerged as a key paradigm, promising to augment the capability of mobile devices through provisioning of computational resources on demand, and enabling resource-constrained mobile devices to offload their processing and storage requirements to the cloud infrastructure. Follow Me Cloud (FMC), in turn, has emerged as a concept that allows seamless migration of services according to the corresponding users’ mobility. Meanwhile, Software Defined Networking (SDN) is a new paradigm that permits the decoupling of the control and data planes of traditional networks and provides programmability and flexibility, allowing the network to dynamically adapt to changing traffic patterns and user demands. Whilst the SDN implementations are gaining momentum, the control plane is still suffering from scalability and performance concerns for a very large network. In this paper, we address these scalability and performance issues in the context of 5G mobile networks by introducing a novel SDN/OpenFlow-based architecture and control plane framework tailored for MCC-based systems and more specifically for FMC-based systems where mobile nodes and network services are subject to constraints of movements and migrations. Contrary to a centralized approach with a single SDN controller, our approach permits the distribution of the SDN/OpenFlow control plane on a two-level hierarchical architecture: a first level with a global controller G-FMCC, and second level with several local controllers LFMCC(s). Thanks to our control plane framework and Network Function Virtualization (NFV) concept, the L-FMCC(s) are deployed on-demand, where and when needed, depending on the global system load. Results obtained via analysis show that our solution ensures more efficient management of control plane, performance maintaining, and network resources preservation.

I. INTRODUCTION

TODAY, service provisioning finds in the emerging Cloud Computing paradigm a flexible and economically efficient solution, in particular for small and medium enterprises that do not want to invest huge capitals for creating and managing their own IT infrastructures. The basic tenet of cloud computing is that end users do not need to care about where a service is actually hosted, while service providers may dynamically acquire the resources they need for service provisioning in a pay-per-use model. While for most of elastic web applications the relative position of client and server end systems does not affect the perceived Quality of Experience, rich interactive applications are sensible to other communication metrics, such as delay and jitter [1]. In the absence of explicit QoS control mechanisms in the network, the only way to improve Quality of Experience is to locate servers as close as possible to user terminals. Such an approach, largely exploited by Content Delivery Networks, can be further advanced in the era of Cloud Computing [2]. Assuming that several cloud-enabled Data Centers are made available at the edges of the Internet (i.e. Federated Cloud), service providers may take advantage of them for optimally locating service instances as close as possible to their users. In such a context, mobility of user terminals makes such location decisions even more difficult. In this context, the Follow Me Cloud (FMC) principle was introduced in [3], wherein mobile users are always connected via the optimal data anchor gateway to access its data and/or service from the optimal DC, i.e. geographically/topologically nearest DC. To ensure an optimal end-to-end connection to the cloud for mobile users, users’ Virtual Machine (VM) (i.e. service) are migrated between DCs, when deemed appropriate[4][5]. Accordingly, services are always provided from data center locations that are optimal for the current locations of the users. It is worth noting that VM migration is seamless and transparent to users. Thus, on-going sessions between users and services are not interrupted and connections do not need to be reestablished, even if users and/or servers (i.e., hosting services) change location. Besides improving users Quality of Service/Quality of Experience, FMC allows preserving operators’ network resources by offloading network traffic to data centers through the nearest points compared with users’ locations.

However, FMC control plane scaling still remains a serious concern in current FMC implementations. To the best of our knowledge, the only work that has investigated the FMC control plane scalability is the one by Bifulco et al. [6]. They studied the scalability of an FMC-based system from a static perspective, and proposed an architecture permitting to distribute the control plane on a number of FMC controllers that are statically located in the networks. Nevertheless, the static number and location of FMC controllers may not be suitable constantly because of the dynamic aspect of network load and traffic patterns over time.
To overcome these limitations, we propose in this paper a novel elastic approach based on a SDN/OpenFlow architecture and a control plane framework tailored for mobile cloud computing systems and more specifically for FMC-based systems where mobile nodes and network services are subject to constraints of movements and migrations. In contrary to centralized approach with single SDN controller, our approach permits to distribute the SDN/OpenFlow control plane on a two-level hierarchical architecture: (i) a first level with a global controller G-FMCC, (ii) and second level with several local controllers L-FMCC(s) deployed on-demand, where and when needed, depending on the network dynamics and traffic patterns.

The remainder of this paper is organized as follows. Section II discusses some related work. In Section III, we present the system description and functioning. Section IV studies the control plane scalability of the system. While the Section V addresses the distributed FMC controller operations, the Section VI provides an analytical evaluation of the solution with results’ discussion. The paper concludes in Section VII.

II. RELATED WORK

A. The FMC concept

The FMC concept was initially proposed in [3]. It was dedicated to the case where all mobility management procedures are handled at the 3GPP domain. In [4], an analytical model is presented to evaluate the performance of the FMC mechanism, while in [5] a Markov-Decision Process (MDP) was introduced for the service migration procedure. In [7], an OpenFlow-enabled implementation of FMC was proposed. The paper describes the components needed to enable FMC, in particular the detection of users’ movements, the decision logic for migrating services and the method for making migration seamless. The authors presented a proof-of-concept of FMC based on VMware (i.e., VMotion for live VM migration), a NOX-based FMC controller and OpenFlow switches. As the latter have to handle multiple per-flow rules, scalability of FMC rules became an issue. A distributed and hierarchical architecture of FMC controllers could be a remedy [6]. In [8], the authors use the concept of identifier/locator separation of edge networks to support service continuity in FMC. Effectively, in case of a VM migration, the old IP address serves as an identifier and the new IP address serves as a locator for the mobile node. Whilst this operation ensures somehow service continuity, it incurs an important overhead for manipulating the locator/identifier values on the edge networks. In [9], the authors proposed another implementation of FMC based on LISP (Local/Identifier Separation Protocol), whereby the main goal is to render FMC independent from the underlying radio access technology. Thanks to the features of LISP, both users’ mobility and VM migration are jointly managed at the same control plane. Besides the LISP entities, all FMC entities were implemented as virtualized network functions running on VMs, facilitating further the concept of carrier cloud [10]. The results obtained from a real-life testbed of the proposed LISP-based FMC architecture showed that the architecture achieved its main design goals, transferring users’ services in the order of milliseconds and with very minimal downtime.

B. SDN Scalability

In the literature, several research efforts have been made to tackle the SDN scalability concerns, most of them can be classified in three categories: data plane, control plane, and hybrid.

In the first category, DevoFlow [11] is characterized by its capability to reduce the overhead by delegating some work to the forwarding devices. Thus, it permits to reduce the control plane invocation for most flow setups, and reduces statistics flows transfer. The Software-Defined Counters (SDC) [12] proposal aims to introduce general-purpose CPUs in forwarding devices (ASIC). The existence of such purpose-CPUs, and a fast connection to ASIC’s data plane allow to replace traditional counters with a stream of rule-match records which is transmitted to and processed in the CPU. Software-defined counters permits to reduce the control plane overhead by allowing software based implementations of functions for data aggregation and compression.

The second category of efforts aims to improve the performance of the control plane. Maestro [13] is an OpenFlow controller which incorporates an abstraction layer that permits to keep the simple single-threaded programming model along with exploiting parallelism, techniques and designs, permitting to improve the performance of the OpenFlow control plane. HyperFlow [14] is another proposal aiming to increase the OpenFlow control plane performance. HyperFlow exploits the distribution of control plane to provide a physical local view and a logical global view of the system. A distributed file system (WheelFS) is used to maintain and synchronize HyperFlow global view state among distributed controllers. Kandoo [15] is a distributed control plane constructed of two-level hierarchical controllers. Local controllers with no interconnection, which take actions of local scope, and global controller that takes actions of global scope requiring global network view.

Among the hybrid category proposals, DIFANE [16] tries to split the control plane between controllers and specialized data plane switches, called authority switches. The latter are responsible for installing rules on the remaining switches, while the controller focuses on generating the needed rules by the applications. The utilisation of this approach ensures a better scale of the overall system.

Additionally, we can distinguish a specific category of proposals that addresses the notion of elasticity in SDN controllers. Elastic approaches aim to include dynamic adaptation of the controllers number and their locations in the design of scalable SDN solutions. In [17] the authors propose ElastiCon, an elastic distributed controller architecture which permits dynamically to expand or shrink the controllers pool according to the network traffic load. A novel protocol of switch migration is also presented permitting to shift traffic across controllers. Authors of [18] presented Pratyaastha, an elastic distributed SDN control plane which permit to efficiently assign state partitions and
switches to controller instances, while minimizing both inter-controller communication and resources consumption. The solution relies on assignment/reassignment algorithm to adapt the system to dynamic changes, which are modeled as an integer linear program (ILP) and solved via a heuristic approach. Authors of [19] proposed a management framework that allows to optimize the controllers’ numbers and locations according to the network dynamics. The dynamic controller provisioning problem was addressed as an integer linear program (ILP) along with two heuristics-based solutions. The presented results are promising and lead to minimize both flow setup time and communication overhead.

III. FMC SYSTEM DESCRIPTION AND FUNCTIONING

The studied FMC system consists of several PMIPv6 domains (we denote this number by $N$), wherein each domain comprises two parts (i) the mobile operator part (ii) and the service provider part (Cloud). The SDN/OpenFlow architecture of the system is constructed of a Global Follow-Me Cloud Controller (G-FMCC), multiple Local Follow-Me Cloud Controllers (L-FMCCs) and a set of OpenFlow-enabled devices which are LMA(s) and DCG(s). First, the system starts with one Global Follow Me Cloud Controller, namely the G-FMCC which is responsible of generating, managing and installing OpenFlow Rules (OFRs) on all PMIPv6 domains’ OpenFlow-enabled devices. It is worth noting that, the number of controllers is dynamically adjusted, while Local Follow-Me Cloud Controllers are dynamically deployed per PMIPv6 domains (one L-FMCC per PMIPv6 domain) according to network dynamics in order to offload the control plane overhead of the global controller G-FMCC.

Our design of Distributed Follow Me Cloud Controller called (DFMCC) is mainly focused on evaluating the scalability in term of managed OpenFlow Rules (OFRs). For this purpose, we introduce a new performance indicator called the OpenFlow Rule Management Rate (OFRRate), which represents the number of new OpenFlow rules managed per second by the OpenFlow controller associated to IPs addresses migrations experienced by a domain $D_i$. We introduced this indicator specifically for FMC-based systems and more generally for mobile cloud computing systems where mobile nodes and network services are subject to movements and migrations. The OFRRate is a domain-declared calculated parameter (denoted by L-OFRRate$_i$ for domain $D_i$) and it can be associated to G-FMCC or L-FMCC (according to the assignment of the domain $D_i$ to G-FMCC or L-FMCC$_i$ at time $t$). If we define a binary vector $Y = \{y_1, y_2, \ldots, y_N\}$ indicating which domains are assigned to L-FMCCs (i.e., $y_m = 1$) and which domains are assigned to G-FMCC (i.e., $y_m = 0$) at any time. In this condition, the sum over the $N$ domains of different L-OFRRate$_i$ values to give the global indicator (denoted by G-OFRRate) of the overall system is as follow:

$$G\text{-OFRRate} = \sum_{i}^{N} L\text{-OFRRate}_i(1 - y_i) \quad (1)$$

The decision of deploying L-FMCC(s) is governed by a threshold-based system. We define two global threshold levels: High Global Threshold (H-GThr) and Low Global Threshold (L-GThr). Our objective is to maintain the system performance in the predefined thresholds window, by deploying L-FMCC(s) when the G-OFRRate goes over H-GThr (G-OFRRate > H-GThr) and remove all deployed L-FMCC(s) to preserve system resources when the G-OFRRate goes under L-GThr (G-OFRRate < L-GThr).

To achieve this, every controller maintains a migration information table, namely Global Migration Information Table (G-MITab) for G-FMCC and Local Migration Information Table (L-MITab) for L-FMCC. The G-MITab is a global view, it contains inter-domain migrations information for all PMIPv6 domains which consists of domain id (Domain), controller id (Controller), number of OFRs (NbrOFR), list of OFRs (ListOFR), OpenFlow rule management rate (OFRRate), and old number of OFRs (OldNbrOFR). For all newly inter-domain migration, the G-FMCC updates accordingly its G-MITab table. Whereas the L-MITab is a local view, it contains inter-domain migrations information only for local PMIPv6 domain with the same attributes as G-MITab.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Controller</th>
<th>NbrOFR</th>
<th>ListOFR</th>
<th>OFRRate</th>
<th>OldNbrOFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_1$</td>
<td>G-fmcc</td>
<td>-</td>
<td>list$_1$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$D_2$</td>
<td>L-fmcc$_2$</td>
<td>-</td>
<td>list$_2$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$D_3$</td>
<td>L-fmcc$_3$</td>
<td>-</td>
<td>list$_3$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$D_N$</td>
<td>G-fmcc</td>
<td>-</td>
<td>list$_N$</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain</th>
<th>Controller</th>
<th>NbrOFR</th>
<th>ListOFR</th>
<th>OFRRate</th>
<th>OldNbrOFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_1$</td>
<td>L-fmcc$_1$</td>
<td>-</td>
<td>list$_1$</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

In order to cope with the problem of scalability and resiliency in centralized control plane architecture, we propose in this work an elastic control plane based on two-level architecture. (i) The first level is represented by the G-FMCC, it is a permanent active controller which is responsible of generating, managing and installing OpenFlow Rules (OFRs) in order to ensure a seamless migration of service on the cloud side, while following inter-domain mobility of MN in the mobile network side. (ii) The second level is represented by the L-FMCC(s) which are dynamically provisioned and deployed when and where needed according to the network dynamics in terms of MNs inter-domains mobilities, services migrations and traffic load. We envision deploying L-FMCC(s) controllers on-demand using the concept of Network Function
Virtualization (NFV) [20] which aims at running network functions in virtualized environments on VMs on top of virtualized platforms, rather than on dedicated hardware. This is expected to help in rapid deployment of FMC solution, at least within the cloud side.

Fig. 1. Number of managed OFRs in MN inter-domain migrations

IV. CONTROL PLANE SCALABILITY

In order to study the performances of our system, we are aiming in this section at assessing the scalability of our distributed control plane architecture. We will mainly focus on evaluating the scalability from the perspective of managed OFRs. Let $c_{jk,i}$ represents the number of correspondent nodes in the cloud side that are exchanging packets with the $i$-th IP address migration from the domain $D_j$ to the domain $D_k$, and $f_{j}$, $f_{k}$ represent the number of OpenFlow-enabled devices present in domain $D_j$ and $D_k$ respectively (LMA for mobile operator side, DCG for cloud provider side). The number of OFRs managed by the G-FMCC for the $i$-th migrated IP address from the domain $D_j$ to the domain $D_k$ is given by the following formula:

$$R_{i}^{D_j,D_k} = (f_{j} + f_{k})c_{jk,i}$$  \hspace{1cm} (2)

The total number of OFRs managed by the G-FMCC for all IPs address migrations from the domain $D_j$ to the domain $D_k$ is the sum over $i$ of the rules as expressed in (2):

$$R^{D_j,D_k} = \sum_{i} (f_{j} + f_{k})c_{jk,i}$$  \hspace{1cm} (3)

The total number of OFRs managed by the G-FMCC for all IPs address migrations originated from the domain $D_j$ is:

$$R^{D_j,D_*} = \sum_{k=1}^{N} \sum_{i \neq j} (f_{j})c_{jk,i}$$  \hspace{1cm} (4)

The total number of OFRs managed by the G-FMCC for all IPs address migrations toward the domain $D_j$ is:

$$R^{D_* ,D_j} = \sum_{k=1}^{N} \sum_{i \neq j} (f_{j})c_{kj,i}$$  \hspace{1cm} (5)

The total number of OFRs managed by G-FMCC for the domain $D_j$ is:

$$R^{D_j} = R^{D_j,D_*} + R^{D_* ,D_j}$$  \hspace{1cm} (6)

$$R^{D_j} = \sum_{k=1}^{N} \left( \sum_{i} (f_{j})c_{jk,i} + \sum_{i \neq j} (f_{j})c_{kj,i} \right)$$  \hspace{1cm} (7)

The total number of OFRs managed by G-FMCC for all domains is the sum over $j$ of the $R^{D_j}$:

$$R^{G} = \sum_{j=1}^{N} \sum_{k=1}^{N} \left( \sum_{i} (f_{j})c_{jk,i} + \sum_{i \neq j} (f_{j})c_{kj,i} \right)$$  \hspace{1cm} (8)

From equations (2), (6) and (7) it is clear that the number of OpenFlow rules generated for a domain $D_j$ is directly proportional to: (i) the number of concurrent inter-domain migrations between the domain $D_j$ and all the other domains (from domain $D_j$ to all other domains or inversely from all other domains to domain $D_j$); (ii) the number of OpenFlow-enabled devices of each domain on which the OpenFlow rules are pushed (represented here by the $f_j$ variables); (iii) the number of correspondent nodes on the cloud side that are exchanging packets with each migrated address (given here by the $c_{jk,i}$ variables) related to the $i$-th IP address migration from the domain $D_j$ to the domain $D_k$.

Due to this characteristic, and in order to quantify the control plane performance in our architecture, we make use of our introduced parameter OFRMRate, which represents the number of new OpenFlow rules managed per second associated to IPs addresses migrations registered in a domain $D_j$. The OFRMRate parameter is a specific characteristic to each domain, it is directly related to the number of migrations experienced by the different domains. OFRMRate is calculated and registered globally on the G-FMCC for all domains. It is also calculated and registered locally if a L-FMCC is deployed for a specific domain. The key objective of our DFMCC solution is to maintain the G-OFRMRate value of the overall system within the prespecified threshold window (H-GThr, L-GThr). This is achieved by the dynamic adaptation of controllers number through the on-demand NFV deployment/removal of L-FMCC(s). The system should deploy one or several L-FMCCs when OFRMRate goes over the H-GThr in order to ensure the offload of the G-FMCC. Inversely, it should remove all deployed L-FMCCs when OFRMRate goes under the L-GThr in order to maintain the system performance and resources preservation.
V. DISTRIBUTED FMC CONTROLLER OPERATIONS

A. Operations Related to Inter-domain Migrations

We point out here the existence of external elements which are the Inter-Domain Mobility Database (IDMD) ensuring the registration of mobility information of all PMIPv6 domains, and the Decision Making Application Module (DMAM) responsible for taking the decision on the relevance of service migration. For each inter-domain migration $M_{D_jD_k}$ from domain $D_j$ to domain $D_k$, the IDMD acts as a trigger to advise the source domain controller $D_j$-FMCC about the MN inter-domain movement. The $D_j$-FMCC exploits these information and thanks to the DMAM module takes a decision on the relevance of service migration and thus triggers the service migration on the cloud side according to user mobility. We note that the $D_j$-FMCC may be, as appropriate, G-FMCC or L-FMCC $j$ depending on the current state of the DFMCC system. The details of control operations interactions between the $D_j$-FMCC and the different modules foregoing the service migration is out-of-scope for this paper the interested readers are invited to refer to our work on centralized FMCC architecture [21] for further details.

In addition, the IDMD maintains a local list of domain-to-controller mapping information indicating at all moment which domain is managed by which controller. This list is kept updated by the G-FMCC according to L-FMCC(s) deployment state. Upon reception of inter-domain migration message $Ms^{D_jD_k}$ from domain $D_j$ to domain $D_k$, the IDMD extracts the source domain $D_j$ of the migration, performs a lookup of its current deployed controller $D_j$-FMCC thanks to the domain-to-controller mapping list, and relays to it the migration message $Ms^{D_jD_k}$. In its turn, the $D_j$-FMCC activates the DMAM in order to take decision on the relevancy of service migration following the MN movement. If the service migration is deemed appropriate, the $D_j$-FMCC generates then the requisite OpenFlow rules in order to ensure a seamless service migration from domain $D_j$ to domain $D_k$ and updates its local table entry (G-MITab or L-MITab depending on the type of current $D_j$-FMCC controller: G-FMCC or L-FMCC $j$)

with information on NbrOFR, and ListOFR. This is achieved by installing the generated OpenFlow rules on all SDN-capable components of $D_j$ and $D_k$ domains.

![Fig. 2. L-FMCC(s) Deployment Conditions](image)

![Fig. 3. (a) Workflow of new migration managed by G-FMCC (b) Workflow of new migration managed by L-FMCC](image)

![Fig. 4. (a) Workflow for L-FMCC(s) deployment (b) Workflow for L-FMCC(s) removal](image)
1) OFRMRate Performance Indicator Computation: 
Let \( \tilde{V} = (\tilde{v}_1, \tilde{v}_2, \ldots, \tilde{v}_N) \) and \( V = (v_1, v_2, \ldots, v_N) \) two vectors which represent respectively the previous OpenFlow rule management rate (OldOFRMRate) and the current OpenFlow rule management rate (OFRMRate) extracted from the G-MITab table of the G-FMCC. Hence, \( \tilde{v}_m \) and \( v_m \) are respectively the previous (OldOFRMRate) and the current (OFRMRate) number of OpenFlow rules managed per second registered for the domain \( D_m \). We assume that the OFRMRate attribute values are updated each \( T_{Rate} \) time interval for all domains in the G-MITab of the G-FMCC. The updated OFRMRate attribute value for the \( D_i \) domain is given as follows:

\[
OFRMRate_i = \frac{NbrOFR_i - OldNbrOFR_i}{T_{Rate}} 
\]  
(9)

In addition, we define a binary vector \( Y = (y_1, y_2, \ldots, y_N) \) indicating which domains have deployed their L-FMCCs (i.e., \( y_m = 1 \)) and which domains are not (i.e., \( y_m = 0 \)) at any time.

2) OFRMRate Updating Algorithm (OUA): This algorithm is invoked by the G-FMCC every \( T_{Rate} \) time interval; it permits to compute the updated value of OFRMRate, indicator for each domain \( D_i \) on the G-MITab. Thereby preparing the next step for the execution of the L-FMCC(s) Deployment Vector Generating Algorithm; which performs the following operations:

<table>
<thead>
<tr>
<th>OFRMRate Updating Algorithm (OUA)</th>
</tr>
</thead>
</table>
| **Input:** Current deployment vector of L-FMCCs, \( Y \)  
Current OpenFlow rule management rate vector, \( V \)  
Previous OpenFlow rule management rate vector, \( \tilde{V} \) |
| **Output:** Updated value of OFRMRate, \( V, \tilde{V} \) |

1. for \( i = 1 \) to \( N \) do  
2. \( \text{if } y_i = 0 \) then  
3. \( \text{G-MITab}[D_i][OFRMRate] \leftarrow \text{G-MITab}[D_i][NbrOFR] - \text{G-MITab}[D_i][OldNbrOFR] \)  
4. \( \tilde{v}_i = \text{G-MITab}[D_i][OFRMRate] \)  
5. \( \text{G-MITab}[D_i][NbrOFR] \leftarrow \text{G-MITab}[D_i][NbrOFR] \)  
6. \( \tilde{v}_i = \text{G-MITab}[D_i][OldNbrOFR] \)  
7. end if  
8. repeat

| TABLE V |
| G-MITAB OFRMRate UPDATE EVERY \( T_{Rate} \) TIME INTERVAL |

<table>
<thead>
<tr>
<th>Domain</th>
<th>Controller</th>
<th>NbrOFR</th>
<th>OldNbrOFR</th>
<th>OFRMRate</th>
<th>OldOFRMRate</th>
</tr>
</thead>
<tbody>
<tr>
<td>( D_1 )</td>
<td>G-fmcc</td>
<td>(-)</td>
<td>(-)</td>
<td>(-)</td>
<td>(-)</td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( D_N )</td>
<td>G-fmcc</td>
<td>(-)</td>
<td>(-)</td>
<td>(-)</td>
<td>(-)</td>
</tr>
</tbody>
</table>

3) L-FMCC(s) Deployment Vector Generating Algorithm (LDVGA): This algorithm is invoked by the G-FMCC every \( T_{deployment} \) time interval (note that \( T_{deployment} > T_{Rate} \)), on the basis of the updated OFRMRate, indicator value for each domain \( D_i \); it permits to generate a candidate deployment vector \( \tilde{Y} = (\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_N) \) of L-FMCC(s) destined to replace the current deployment vector \( Y = (y_1, y_2, \ldots, y_N) \), each one in its correspondent domain. This algorithm prepares the NFV deployment of L-FMCC(s) step accomplished by L-FMCC(s) NFV Deployment Algorithm, and it performs the following operations:

<table>
<thead>
<tr>
<th>L-FMCC(s) Deployment Vector Generating Algorithm (LDVGA)</th>
</tr>
</thead>
</table>
| **Input:** Current OpenFlow rule management rate vector, \( V \)  
Current deployment vector of L-FMCCs, \( Y \) |
| **Output:** Candidate deployment vector of L-FMCCs, \( \tilde{Y} \) |

1. \( Y^* \leftarrow V \), with \( v_m \) sorted in descending order  
2. \( \tilde{Y}^* \leftarrow Y, \) with \( y_m \) sorted in the same index order as \( v_m \)  
3. \( \psi \leftarrow \sum_{i=1}^{N} v_i^* (1 - y_i^*) \), the G-OFRMRate  
4. \( \text{if } \psi > H-GThr \) then  
5. \( \psi^* \leftarrow \psi \)  
6. \( \text{for } i = 1 \) to \( N \) do  
7. \( \text{if } y_i^* = 0 \) then  
8. \( \psi^* \leftarrow \psi^* + v_i \)  
9. \( y_i^* \leftarrow 1 \)  
10. end if  
11. if \( \psi^* \leq H-GThr \) or \( i = N \) then  
12. break  
13. end if  
14. repeat  
15. end if  
16. if \( \psi < L-GThr \) then  
17. \( \psi^* \leftarrow \psi \)  
18. \( \text{for } i = 1 \) to \( N \) do  
19. \( \text{if } y_i^* = 1 \) then  
20. \( \psi^* \leftarrow \psi^* + v_{N-i+1} \)  
21. \( y_{N-i+1}^* \leftarrow 0 \)  
22. end if  
23. if \( \psi^* \geq L-GThr \) or \( i = N \) then  
24. break  
25. end if  
26. repeat  
27. end if  
28. \( \tilde{Y} \leftarrow Y^* \), with \( y_m \) sorted in the same index order as \( y_m \)

4) L-FMCC(s) NFV Deployment Algorithm (LNDGA): This algorithm is triggered by the L-FMCC(s) Deployment Vector Generating Algorithm, marking the end of its execution. It is launched by the NFV module of the G-FMCC, based on the current deployment vector \( Y \) and the candidate deployment vector \( \tilde{Y} \). It permits to deploy/remove L-FMCC(s) in order to adapt the overall system load. The algorithm performs the following operations:

<table>
<thead>
<tr>
<th>L-FMCC(s) NFV Deployment Algorithm (LNDGA)</th>
</tr>
</thead>
</table>
| **Input:** Current deployment vector of L-FMCCs, \( Y \)  
Candidate deployment vector of L-FMCCs, \( \tilde{Y} \) |
| **Output:** New deployment vector of L-FMCCs, \( \tilde{Y} \) |

1. for \( i = 1 \) to \( N \) do  
2. \( \text{if } y_i = 0 \) and \( \tilde{y}_i = 1 \) then  
3. Deployment of L-FMCC in the domain \( D_i \) with NFV  
4. \( \text{G-MITab}[D_i][Controller] \leftarrow \text{L-FMCC} \)  
5. \( \text{IDMD domain-to-controller list}[D_i] \leftarrow \text{L-FMCC} \)  
6. Transfer of context: \( \text{L-MITab}[D_i] \leftarrow \text{G-MITab}[D_i] \)  
7. end if  
8. \( \text{if } y_i = 1 \) and \( \tilde{y}_i = 0 \) then  
9. Removal of L-FMCC in the domain \( D_i \) with NFV  
10. \( \text{L-MITab}[D_i][Controller] \leftarrow \text{G-FMCC} \)  
11. \( \text{IDMD domain-to-controller list}[D_i] \leftarrow \text{G-FMCC} \)  
12. Transfer of context: \( \text{G-MITab}[D_i] \leftarrow \text{L-MITab}[D_i] \)  
13. end if  
14. repeat  
15. \( Y \leftarrow \tilde{Y} \), candidate vector becomes the current deployment vector
VI. Evaluation

In this section, we present the evaluation of our Distributed Follow Me Cloud Controller DFMCC through a theoretical analysis. With regard to the scalability of the distributed architecture and to evaluate the total number of managed rules we mainly focus on the formulas given in (7) and (8). In order to simulate the inter-domain migration arrivals for a domain \(D_j\), we rely on Non-Homogeneous Poisson Process (NHPP) model with rate parameter function \(\lambda_jk(t)\). Thus in the studied scenarios the inter-domain migrations arrivals are assumed to follow a Non-Homogeneous Poisson Process with rate parameter function \(\lambda_jk(t)\).

A. Non-Homogeneous Poisson Process Model for Inter-domain Migrations Arrivals

In this scenario the inter-domain migrations arrivals for a given \(t\) from domain \(D_j\) to domain \(D_k\) (noted \(N_{D_jD_k}(t)\)) are assumed to follow Non-Homogeneous Poisson Process with rate parameter function \(\lambda_jk(t)\); i.e.,

\[
\begin{align*}
\begin{cases}
P(N_{D_jD_k}(t) = R) = e^{-\Lambda_jk(t)}(\Lambda_jk(t))^R/R!, & t \geq 0 \\
\Lambda_jk(t) = \int_0^t \lambda_jk(s) \, ds
\end{cases}
\end{align*}
\]

(10)

In our evaluation we will consider three different scenarios, each scenario is run for 30 minutes with the L-FMCC(s) Deployment Vector Generating Algorithm (LDVGA) running every 4 minutes \((T_{deployment} = 4\,\text{min})\). The choice of the latter depends on the current state of the DFMCC system, and can be further tuned through a more detailed analysis. Three scenarios were considered: (1) The first scenario is the growing phase in which the inter-domain migrations arrivals are assumed to increase; (2) The second scenario represents a constant phase with inter-domain migrations arrivals assumed to be constant; (3) The third scenario is the decaying phase in which the inter-domain migrations arrivals are assumed to decrease. In order to meet the conditions of this three scenarios: the growing phase, the constant phase and the decaying phase, the rate parameter function \(\lambda_jk(t)\) from domain \(D_j\) to domain \(D_k\) is given by:

\[
\lambda_jk(t) = \begin{cases} 
\frac{t}{5(j+1)}, & \text{if } 0 \leq t \leq 30mn \\
\frac{360}{(j+1)}, & \text{if } 30mn \leq t \leq 60mn \\
\frac{360}{(j+1)(1+\frac{t}{100})}, & \text{if } 60mn \leq t \leq 90mn 
\end{cases}
\]

Where \(t\) is time and \(j\) is the index of the source domain \(D_j\).

The expected number of inter-domain migration arrivals \(M_{D_jD_k}^D(t)\) from domain \(D_j\) to domain \(D_k\) for the NHPP \(N_{D_jD_k}(t)\) is given by:

\[
M_{D_jD_k}^D(t) = E[N_{D_jD_k}(t)] = \Lambda_jk(t)
\]

(11)

As a result,

\[
M_{D_jD_k}^D(t) = \begin{cases} 
\frac{t^2}{10(j+1)}, & \text{if } 0 \leq t \leq 30mn \\
360t, & \text{if } 30mn \leq t \leq 60mn \\
36000 \log(1+\frac{t}{100}), & \text{if } 60mn \leq t \leq 90mn
\end{cases}
\]

Throughout our evaluation, the number of domains \(N\) is fixed to \(6\) \((N = 6)\) and the reference inter-domain migrations scheme is represented in Figure 5. We consider also that the \(c_{j,k,i}\) values are constant and equal to \(c\) value whatever the \(i\)-th inter-domain migration as well as the \(D_j\) and \(D_k\) domains \((\forall i,j,k, c_{j,k,i} = c)\). Moreover, the number of OpenFlow-enabled devices \(f_j\) are constant and equal to \(f\) value whatever the \(D_j\) domain \((\forall j,f_j = f)\).

![Network topology of DFMCC architecture](image)

Fig. 5. Network topology of DFMCC architecture

Regarding the preceding criteria and the formulas given in (7) and (8), we will obtain the equations summarized in Table VI:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Domain (D_j)</th>
<th>All domains</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFRMRate</td>
<td>(L)-OFRMRate(<em>{ij}(t)) = (2f_i\sum</em>{k=1}^{N} c_{j,k,i} \Lambda_{jk}(t))</td>
<td>G-OFRMRate(<em>{ij}(t)) = (2f_i\sum</em>{k=1}^{N} c_{j,k,i} \Lambda_{jk}(t))</td>
</tr>
<tr>
<td>#Migrations</td>
<td>(M^{D_j}(t) = \sum_{k \neq j} c_{j,k,i} \Lambda_{jk}(t))</td>
<td>(M^{D_k}(t) = \sum_{j=1}^{N} c_{j,k,i} \Lambda_{jk}(t))</td>
</tr>
<tr>
<td>#Rules</td>
<td>(R^{D_j}(t) = 2f_i M^{D_j}(t))</td>
<td>(R^{D_k}(t) = 2f_i M^{D_k}(t))</td>
</tr>
</tbody>
</table>

B. scenario 1: OFRMRate increasing phase

This scenario is characterized by an increasing number of inter-domain migrations arrivals according to the \(M^{D_j}(t)\) function shape of the NHPP model of the domain \(D_j\). Figure 6(a) plots the global G-OFRMRate\(_{ij}(t)\) associated to the G-FMCC and the L-OFRMRate\(_{ij}(t)\) associated to the L-FMCC\(_i\) of domain \(D_i\) when this latter is activated. We can see that the application of our elastic control plane framework permits to offload the G-FMCC when the G-OFRMRate\(_{ij}(t)\) goes over the G-HThr by the L-FMCC\(_i\) NFV deployment for most loaded domain\(_i\). The Figure 7(a) plots a comparison of the global number of rules managed by the G-FMCC under the...
application of our elastic control plane framework, and the case with a single centralized FMCC. We can see that the G-FMCC becomes less loaded then the centralized FMCC when the G-HThr is reached.

C. scenario 2: OFRMRate stationary phase

This phase is called stationary because the number of inter-domain migrations arrivals \( M_{Dj}(t) \) grows according to a particular kind of NHPP with function parameter a constant (which is a Homogeneous Poisson Process HPP). In Figure 6(b) we can see that the application of our solution always ensures to have a G-OFRMRate(t) under the G-HThr, and as this phase is stationary (number of inter-domain migrations arrivals is constant) only the first invocation of our LDVGA and LNDA algorithms (the first \( T_{deployment} \)) is needed to deploy the sufficient number of L-FMCC(s) ensuring a G-OFRMRate(t) under the G-HThr for the entire duration of this phase. As we can see also in the Figure 7(b) there is only the first execution of our algorithms (the first \( T_{deployment} \)) that permits to reduce the number of OpenFlow rules managed by the G-FMCC. We can clearly distinguish differences in term of number of managed rules and the advantage provided by our solution.

D. scenario 3: OFRMRate decreasing phase

This scenario is characterized by a decreasing number of inter-domain migrations arrivals according to the \( M_{Dj}(t) \) function shape of the NHPP model of a domain \( D_j \). Figure 6(c) plots the global G-OFRMRate(t) associated to the G-FMCC and the L-OFRMRate(t) associated to the L-FMCC of domain \( D_i \) when the latter is activated. We can see that the application of our elastic approach permits this time to load the G-FMCC when the G-OFRMRate(t) goes under the G-LThr by the L-FMCC(s) NFV removing of less loaded domain(s) and their assignment to the G-FMCC. The Figure 7(c) plots a comparison of the global number of rules managed by the G-FMCC under the application of our elastic control plane framework, and the case with single centralized FMCC. We can observe clearly that our solution permits to preserve resources when the G-LThr is reached by deactivating L-FMCC(s) and approaching thus the case of centralized FMCC architecture.

E. Network delay

In this section we will analyse the delay of our approach in terms of the number of \( d_r \) exchanged messages. The \( d_r \) is a regional long distance message, which is exchanged between two different domains. Accordingly, this kind of message experiences high delay compared with the \( d_l \) message, as the latter is a local short distance message.

VII. Conclusion

In this paper, we have proposed our design of an elastic distributed SDN controller tailored for mobile cloud computing and FMC-based systems. We presented the building blocks of our control plane framework: the performance indicator OFRMRate and the three algorithms (OUA, LDVGA and LNDA). The evaluation results obtained via analysis show that our solution ensures better control plane management, performances maintaining and network resources preservation.
ACKNOWLEDGEMENT

This work was partially supported by the French FUI’18 DVD2C project.

REFERENCES


Abdelkader Assioui is working toward the Ph.D. degree at the University of Versailles UVSQ, France. He received an M.Sc. in telecommunication and multimedia networking from the University of Versailles. He obtained his Ph.D. degree in computer science from the University of Cergy-Pontoise in 2005, with a dissertation on QoS provisioning in IEEE 802.11-based networks. His other interests include: future Internet networks, mobile networks, QoS, QoE, performance evaluation and multimedia transmission.

Dr. Adlen Ksentini (SM’14) is an Associate Professor at the University of Rennes 1, France. He is a member of the INRIA Rennes team Dionysyos. He received an M.Sc. in telecommunication and multimedia networking from the University of Versailles. He obtained his Ph.D. degree in computer science from the University of Cergy-Pontoise in 2005, with a dissertation on QoS provisioning in IEEE 802.11-based networks. His other interests include: future Internet networks, mobile networks, QoS, QoE, performance evaluation and multimedia transmission.

Mourad Gueroui is an associate professor in computer engineering at the University of Versailles, France. His research interests are in the area of wireless networks (WATM, WLAN and sensors), particularly performance evaluation and QoS provisioning. He received his M.Sc. degree from University of Paris6 and his Ph.D. in networking and computer engineering from the University of Versailles.
Prof. Tarik Taleb is an IEEE Communications Society (ComSoc) Distinguished Lecturer and a senior member of IEEE. He is currently a Professor at the School of Electrical Engineering, Aalto University, Finland. Prior to his current academic position, he was working as Senior Researcher and 3GPP Standards Expert at NEC Europe Ltd, Heidelberg, Germany. He was then leading the NEC Europe Labs Team working on R&D projects on carrier cloud platforms, an important vision of 5G systems. He was also serving as technical leader of the main work package, Mobile Core Network Cloud, in EU FP7 Mobile Cloud Networking project, coordinating among 9 partners including NEC, France Telecom, British Telecom, Telecom Italia, and Portugal Telecom. Before joining NEC and till Mar. 2009, he worked as assistant professor at the Graduate School of Information Sciences, Tohoku University, Japan, in a lab fully funded by KDDI, the second largest network operator in Japan. From Oct. 2005 till Mar. 2006, he worked as research fellow at the Intelligent Cosmos Research Institute, Sendai, Japan. He received his B. E degree in Information Engineering with distinction, M.Sc. and Ph.D. degrees in Information Sciences from GSIS, Tohoku Univ., in 2001, 2003, and 2005, respectively. Prof. Taleb’s research interests lie in the field of architectural enhancements to mobile core networks (particularly 3GPP’s), mobile cloud networking, network function virtualization, software defined networking, mobile multimedia streaming, inter-vehicular communications, and social media networking. Prof. Taleb has been also directly engaged in the development and standardization of the Evolved Packet System as a member of 3GPP’s System Architecture working group. Prof. Taleb is a member of the IEEE Communications Society Standardization Program Development Board. As an attempt to bridge the gap between academia and industry, Prof. Taleb founded the ”IEEE Workshop on Telecommunications Standards: from Research to Standards”, a successful event that got awarded “best workshop award” by IEEE Communication Society (ComSoC). Based on the success of this workshop, Prof. Taleb has also founded and has been the steering committee chair of the IEEE Conf. on Standards for Communications and Networking. Prof. Taleb is the general chair of the 2019 edition of the IEEE Wireless Communications and Networking Conference (WCNC’19) to be held in Marrakech, Morocco. He is/was on the editorial board of the IEEE Transactions on Wireless Communications, IEEE Wireless Communications Magazine, IEEE Transactions on Vehicular Technology, IEEE Communications Surveys & Tutorials, and a number of Wiley journals. He is serving as chair of the Wireless Communications Technical Committee, the largest in IEEE ComSoC. He also served as Vice Chair of the Satellite and Space Communications Technical Committee of IEEE ComSoc (2006 - 2010). He has been on the technical program committee of different IEEE conferences, including Globecom, ICC, and WCNC, and chaired some of their symposia. Prof. Taleb is the recipient of the 2009 IEEE ComSoc Asia-Pacific Best Young Researcher award (Jun. 2009), the 2008 TELECOM System Technology Award from the Telecommunications Advancement Foundation (Mar. 2008), the 2007 Funai Foundation Science Promotion Award (Apr. 2007), the 2006 IEEE Computer Society Japan Chapter Young Author Award (Dec. 2006), the Niwa Yasujiro Memorial Award (Feb. 2005), and the Young Researcher’s Encouragement Award from the Japan chapter of the IEEE Vehicular Technology Society (VTS) (Oct. 2003). Some of Prof. Taleb’s research work have been also awarded best paper awards at prestigious conferences.