



Budapest University of Technology and Economics
Doctoral School of Informatics

Towards Cloud-native Mobility Management

SUMMARY OF THE PH.D. DISSERTATION

Ákos Leiter

RESEARCH SUPERVISOR: László Bokor Ph.D.
BUDAPEST, 2025

Contents

- Abbreviations 3**
- Introduction..... 4**
 - 1.1 Research Objectives..... 5
 - 1.2 Research Methodologies 6
- New Results 7**
 - 1.3 Thesis Group 1: Flow-based operator-centric dynamic mobility management architecture by Proxy Mobile IPv6 7
 - 1.3.1 Architectural design and signalling proposal..... 7
 - 1.3.2 Evaluation results..... 8
 - 1.4 Thesis Group 2: Cloud-native Mobile IPv6 and Proxy Mobile IPv6 14
 - 1.4.1 Architectural design for Cloud-native Mobile IPv6 and Cloud-native Proxy Mobile IPv6 14
 - 1.4.2 Overhead conclusions in a cloud environment for PMIPv6 15
 - 1.4.3 Microservice-based design for Cloud-native Mobile IPv6 Home Agent 16
 - 1.5 Closed loop orchestration procedures for Cloud-native Mobile IPv6 23
 - 1.5.1 Functional and performance evaluation of closed-loop orchestration for Cloud-native Mobile IPv6 Home Agent 23
 - 1.5.2 Numerical calculation 25
- Application and Conclusion 29**
- My Publications..... 30**
- References 32**

Abbreviations

ANDSF	Access and Network Discovery Selection Function
BA	Binding Acknowledgement
BU	Binding Update
CFR	Current Flow Information Request
CNF	Cloud-native Network Function
CN-MIPv6	Cloud-native Mobile IPv6
CN-PMIPv6	Cloud-native Proxy Mobile IPv6
CoA	Care-of-Address
FUR	Flow Update Request
HA	Home Agent
HDR	Handover Discover Request
LMA	Local Mobility Anchor
MAG	Mobile Access Gateway
MIPv6	Mobile IPv6
MN	Mobile Node
MPU	Mobility Policy Update
MTBF	Mean Time Between Failure
MTTR	Mean Time to Repair
MUA	Mobility Update Accept
MUR	Mobility Update Request
NFV	Network Function Virtualization
NPoA	Network Point-of-Attachment
O&M	Operation and Management
ONAP	Open Network Automation Platform
PBA	Proxy Binding Acknowledgement
PBU	Proxy Binding Update
PCRF	Policy and Charging Rule Function
PMIPv6	Proxy Mobile IPv6
PoA	Point-of-Attachment
RA	Router Advertisement
RS	Router Solicitation
SDN	Software Defined Networks
TA	Tracking Area
UE	User Equipment

Introduction

Due to the ever-increasing traffic and mobility events, telecommunication networks are continuously changing. Network Function Virtualisation (NFV), cloud, and Software Defined Networking (SDN) have completely changed the landscape of telecommunication networks in several ways. Mobility management is not an exemption from the trends mentioned above. In this dissertation, I focus on IP-based mobility management, whose main goal is to ensure session continuity with IPv6 address preservation for nomadic users when they change their Point-of-Attachment (PoA) to the network. For this purpose, I utilize the Mobile IPv6 protocol family (xMIPv6, MIPv6)[1] whose further derivative is Proxy Mobile IPv6 (PMIPv6)[2]. Note that this type of mobility management operates in TCP/IP Layer 3; meanwhile, commonly considered handover management operates in Layers 1 and 2 (L1,L2), e.g., changing base stations to continue communication through the network. MIPv6 is user-centric; meanwhile, PMIPv6 is an operator-centric approach, differentiating whether Mobile Nodes (MN) are involved in mobility management. In the case of ordinary PMIPv6, MNs do not know even whether they are mobility-managed. This implies that the usage of PMIPv6 does not require end-user modification, but in the case of MIPv6, MNs need changes.

As the IP layer is independent of the underlying communication stack, thus xMIPv6 can support inter-technology handover, e.g., changing Point of Attachment (PoA) between 3GPP interfaces and IEEE Wi-Fi. This also leads to traffic offloading: if a Wi-Fi network is available, sessions can be moved from public mobile network radio interfaces to the locally available wireless network.

But not all users need mobility management every time. Monitoring and handling these situations lead to the area of Dynamic Mobility Management (DMM). As every mobility event requires additional signalling to deal with, significant control message overhead may arise due to unnecessary signalling.

One of the natural extensions of Dynamic Mobility Management is a session-aware mobility management. MIPv6 and PMIPv6 have their protocol extensions for this purpose: Flow Bindings [3] [4]. 5-tuple-based session identification and management can be ensured where per-flow routing decisions can be executed (a special form of policy routing). In the case of PMIPv6 Flow Bindings, MNs do need modifications opposite to the ordinary PMIPv6. Based on these new advancements, intelligent per-flow traffic offload can be envisioned when multiple networks are available.

The first thesis group of my dissertation is to give a new method, including architectural design for integrating dynamic, flow-aware, operator-centric IP-based mobility management into 3GPP-defined networks. I have created a testbed for evaluating mobility scenarios. Furthermore, I have designed and executed a simulation in MATLAB to

conclude whether dynamic mobility management should be used or not, highlighting signalling cost spare and overhead.

The second thesis group of this dissertation is about designing a cloud-native Home Agent (HA) for MIPv6 to cover the expectation of the latest technology trends at the time of writing those papers. This also leads to the design of a cloud-native Local Mobility Anchor (LMA) and Mobile Access Gateway (MAG) for PMIPv6. I will present numerical results and evaluate containerization cost on a prototype implementation.

The third thesis group covers the questions of software availability by orchestration. Microservice design and the advent of cloud computing have raised the level of complexity by increasing the number of atomic and standalone software components. Handling these new scenarios requires a new element: the orchestrator. Orchestration is the central component of today's telecommunication networks which is responsible for putting the right amount of resources in the right place at the right time. In the context of Mobile IPv6, I have designed and verified a new procedure for automatic failover and scaling scenarios. Based on the results, I calculated whether availability could be increased by automatic failover or not, which can also be considered redundancy restoration time. As an end goal: the quicker the failover is, the less resources are needed to keep the same availability.

1.1 Research Objectives

My essential objective was to create and verify an intelligent traffic offloading scheme with 3GPP networks. Wi-Fi networks means cheaper access to the Internet compared to 3GPP-based interfaces. If we can identify those application in mobile end-users 'apparatus which can be offloaded to Wi-Fi networks, it can save radio resource on base station. But these offload procedures can be more advanced if mobility management is applicable to ensure session continuity. Furthermore, the nomadicity of end-users may be different, so I have used a mathematical model with which the advantage of traffic offload can be examined [5]. This analytical model differentiates between two scenarios: 1) UE moves into the coverage Wi-Fi access point and terminates its traffic, 2) UE moves through a Wi-Fi access point and connects back to the macrocell. It also considers the proportion of high and low mobility users to get the pedestrian and car scenarios as well. The signalling cost of my approach has been built into this model with my extension to cover the ratio of traffic which requires dynamic mobility management.

This leads to the second topic of dissertation, which is the cloudification of IPv6-based mobility management protocols. Due to the new execution environments such as virtualisation, cloud controlled by orchestrators, mobility management protocols cannot be an exemption to adapt to the changing environment. So, if we want to have intelligent traffic offload accelerated by mobility management in the new context, it is needed to examine these protocols in the cloud environment. This is why cloud-native Mobile IPv6

(CN-MIPv6) and cloud-native Proxy Mobile IPv6 (CN-PMIPv6) have been proposed on architectural and implementation level whether and how they can follow microservice designs and the overall cloud concepts as well [C1][C2].

At the end of the research, an actual orchestrator was examined: Open Network Automation Platform (ONAP) [6] how it can work together with CN-MIPv6 and what advantages it can add to the system (e.g.: lifecycle management, increased redundancy, automatic failover and scaling). These new procedures emerge naturally from the cloud execution environments, but their usage may be different for each networking software. There is also a mathematical calculation - based on measurement results of redundancy restoration time– how availability is affected by the fast restoration of network service. The results show that the time of automatic redundancy restoration can be so little which leads to decrease number of redundant nodes. That means cost saving meanwhile the overall availability is not jeopardised. This does not only pertain to CN-MIPv6, but it can also show the power of network automation and cloud execution environments in general.

1.2 Research Methodologies

All the thesis groups uses well-known statistical and analytical toolsets. Mobility and handover probability model is based on Haijun Zhang et al, 's work [5] which I have implemented in MATLAB.

Prototype implementations for CN-MIPv6 was implemented in C programming language with Python extension which has easier support for API calls implementation. Kubernetes is the main orchestration and execution environment where the prototypes were implemented into. These prototypes follow microservice design principles that means Docker container-based software which are scheduled and collocated as Kubernetes Pods. Numerical evaluations are done via well-knows statistical concepts e.g.: minimum, maximum, average, median, standard deviation etc.

Availability calculations are based on well-known software availability tools in the context of telecommunication, detailed in [7] such as: Mean Time Between Failure (MTBF), Mean Time to Repair (MTTR). The diagrams and calculations were implemented with Python programming language. ONAP was used as an orchestrator for getting real measurements to feed the Python-based algorithms. ONAP is mainly based on JAVA and Python languages.

New Results

1.3 Thesis Group 1: Flow-based operator-centric dynamic mobility management architecture by Proxy Mobile IPv6

3GPP networks lack of full integration of Proxy Mobile IPv6 in connection with application-specific routing where Dynamic Mobility Management is also considered. This means operator-centric optimization of IPv6-based mobility management with Flow Bindings is not possible. In this thesis group, I present an architectural design for integrating Proxy Mobile IPv6 and 3GPP networks to support intelligent, application-specific route selection where the Service Provider (SP) is in charge of the decision. Furthermore, there are also measurements and mathematical models for identifying the performance and applicability limits of the solution.

1.3.1 Architectural design and signalling proposal

Thesis 1.1: *I have proposed a new architecture for flow-based operator-centric mobility management in the context of 3GPP networks with Proxy Mobile IPv6. The novel architecture provides a new IPv6-based interface, connections between core network elements and new feature proposals to support intelligent traffic offload. I have introduced novel method in the proposed architecture to initialize traffic offload to the Wi-Fi networks from 3GPP radio interfaces by Proxy Mobile IPv6. The solution is compatible with existing IETF-standardized operation but adds new signaling elements to the 3GPP-maintained area [J1]*

Figure 1 depicts my integration approach, showing new interfaces as requirements for a fully functioning integration. In my proposal, I consider Policy Control and Charging Function (PCRF) as a central intelligent point for deciding about offload execution. But to have the decision executed, it requires several data sources, new interfaces between PCRF and several elements (depicted in Figure 1 in red).

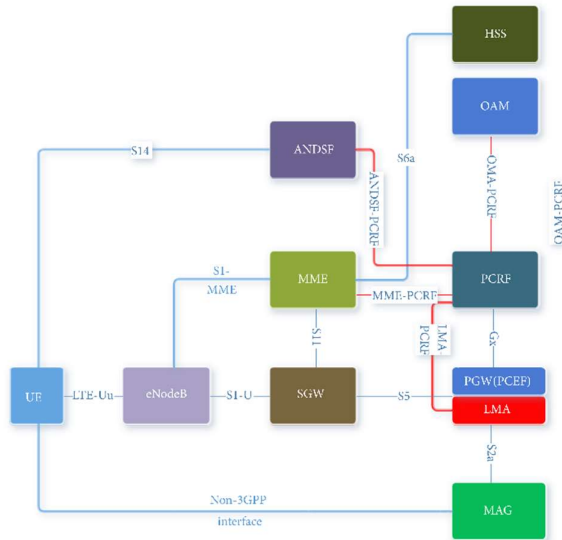


Figure 1 - Proposed architecture for integrating 3GPP network with flow-based based PMIPv6

1.3.2 Evaluation results

Thesis 1.2: *Based on the performed measurements on the implemented model, I have concluded the limits for the number of network changes for TCP sessions in a particular timeframe where the TCP throughput starts to degrade. I have also concluded the limits for the number of network changes for UDP packet flows: UDP packet loss ratio correlates linearly to the number of network changes. [J1]*

The main finding of this Thesis is the following (Figure 2). In cases where there are no handovers, the system behaves even worse than in the case of four network changes, meaning that the offloading could increase throughput even when the ping-pong effect occurs. 3 PoA changes are the maximum during 1 minute where the overall TCP throughput is not degraded.

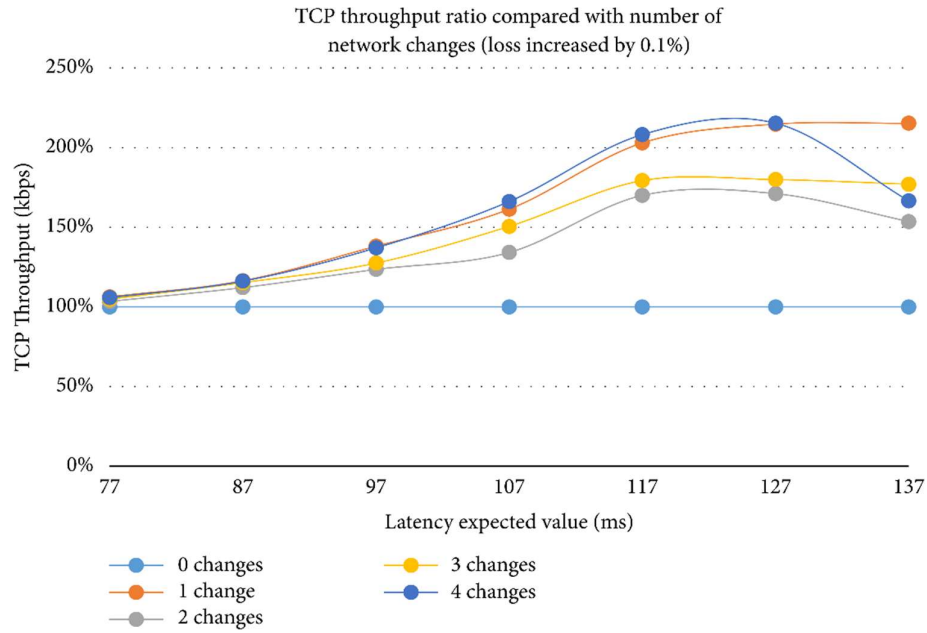


Figure 2 - TCP throughput ratio compared with the number of network changes

In those cases where nothing happens with the flow, the worst-case scenario occurs. Even having four handovers within one minute is better than leaving flow alone (untouched) within degrading transmission circumstances on the 3GPP link.

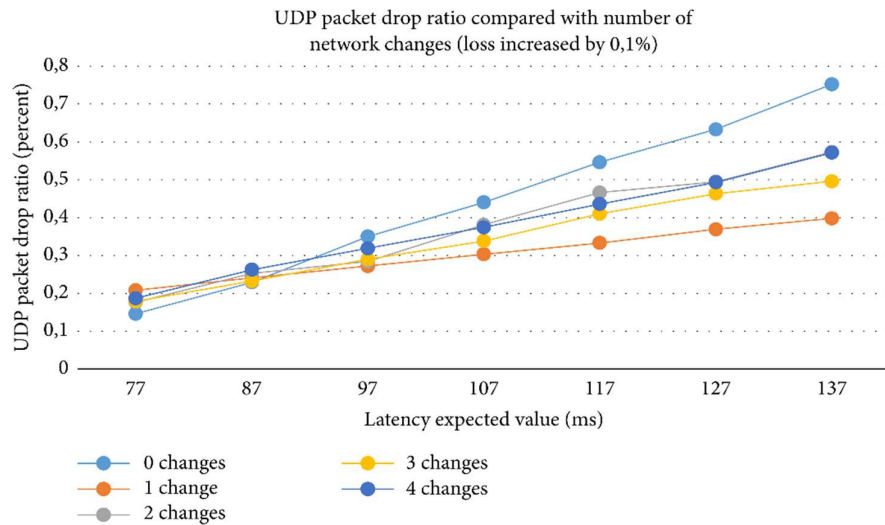


Figure 3 - UDP packet drop ratio compared with the number of network changes

Thesis 1.3: *Based on the performed measurements on the implemented model, I have shown that the tunneling overhead of my proposal is marginal in the case of TCP and UDP user plane traffic. [J1]*

Figure 4 and Figure 5 depict the results, which show marginal overhead for tunnel usage. This gives estimations for the usage of dynamic mobility management as it operates with on-demand tunnel usage.

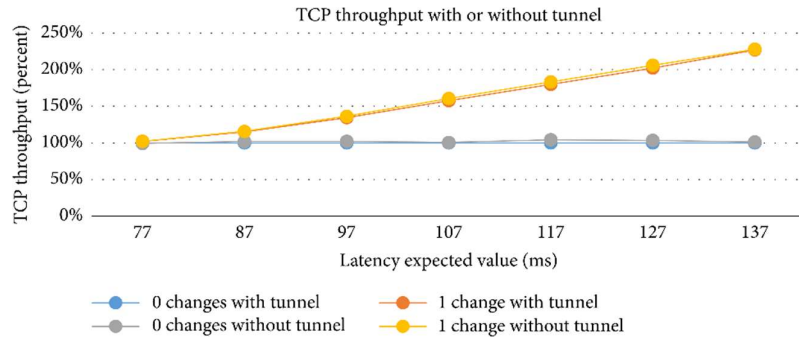


Figure 4 - Results of tunneling overhead in the case of TCP

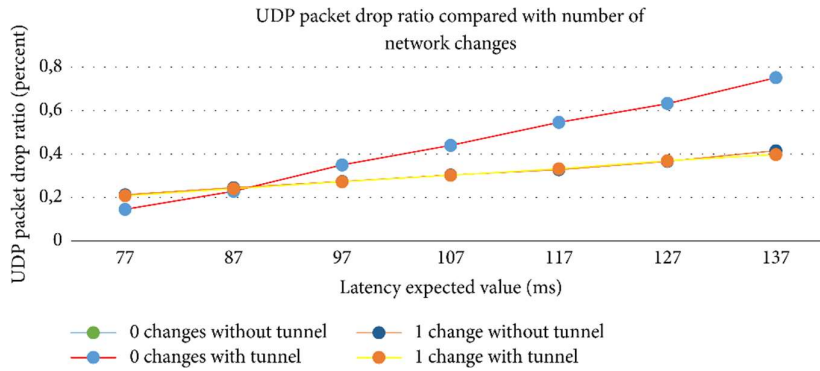


Figure 5 – Results of tunneling overhead in the case of UDP

Theses 1.4: *I have described the signaling overhead characteristics in different mobility cases. The biggest benefit comes when the ratio of nonvehicular users is high, but the number of sessions requiring mobility handling is low. [J1]*

I have considered two scenarios based on [5], depicted Figure 6, Figure 7. Either a user moves through a Wi-Fi network or terminates its session inside it.

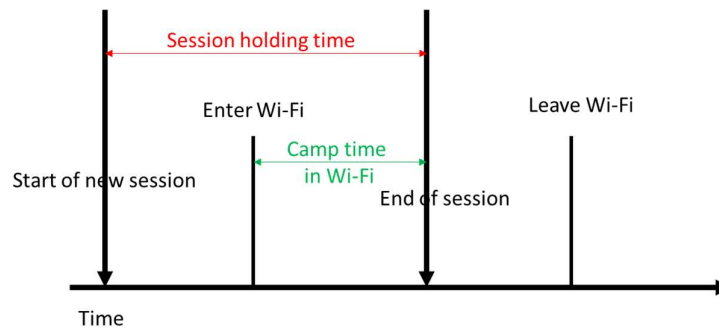


Figure 6 - Scenario #1 - user stops its session in the Wi-Fi

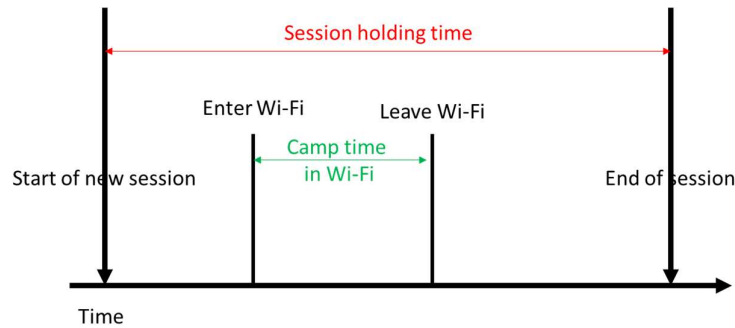


Figure 7 - Scenario #2 user does not stop its session in Wi-Fi

The relevant part of this model for my work which are the inputs as well: there are two types of users considered low mobility (non-vehicular, α) and high-mobility (vehicular users, $1 - \alpha$) on their model. The authors assume the macrocell camp time as exponential distribution for both types of users. The radius of the macrocell (R , $100r$) is considered 100 times then the radius of the Wi-Fi (r). I added a new variable from H. Zhang et al. 'work [5]. I consider the ratio of applications flows (IP flows) which requires mobility management to consider dynamicity.

A following pseudocode introduces the most important part of my MATLAB code:

```
# ratio of vehicular and non-vehicular users
LOOP from alfa=0 to 1 by 0.1
    CALCULATE handover probability for the two scenarios
    #ratio of sessions which requires dynamic mobility
    management in the following loop
    LOOP from 10 to 90 by 40
        CALCULATE control message overhead by actual
        vehicular user ratio, handover probability and
        mobility demand
    END of LOOP
END of LOOP
```

Figure 8 shows a diagram created using the models and calculations based on [5], to show the probability of handover based on the ratio of vehicular and nonvehicular users. The scenario sets a Wi-Fi inside a macrocell: a user goes through it and then terminates its session inside the Wi-Fi. A vehicular user can be considered a car, while a nonvehicular user is usually a pedestrian. The model shows that if a pedestrian goes through the macrocell-Wi-Fi path, it is more likely to have a handover as the network has time to drop the user from the macrocell to the Wi-Fi to offload the macrocell. There are two corner cases which can also be seen in the Figure 8 diagram: even though every user is a pedestrian, that does not mean 100% have a handover; on the contrary, when everyone is a fast mover, who can just run through this macrocell-femtocell scenario, it is not 0% not having any handover, as the network can decide for some reason to put some users into the femtocell even for a short time. Another corner case is when a user moves around

a Wi-Fi cell circularly, even though that user can have high speed, it does not mean it has a handover. Furthermore, the user can be stationary user: even though it does not move, it can have a handover to the macrocell due to the availability of the Wi-Fi network or radio signal degradation etc.

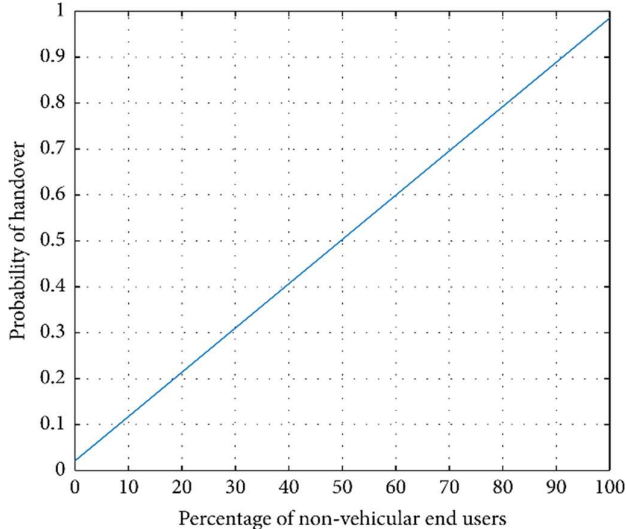


Figure 8 - Probability of handover based on the ratio of vehicular and nonvehicular users.

I evaluated a scenario when 1,000,000 users are moving with 30 sessions per each, and I counted the number of Proxy Binding Update (PBU)/Proxy Binding Acknowledgement (PBA) messages concerning the ratio of vehicular and nonvehicular UEs. The x-axis shows the ratio of user sessions that require mobility handling (i.e., the ratio of user sessions causing mobility control message overhead). The sizes of the PBU and PBA messages are considered 76 bytes.

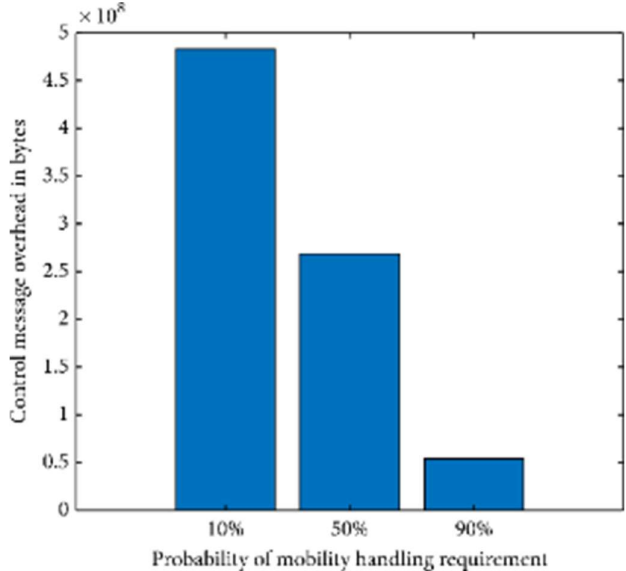


Figure 9 - Control message overhead in case of 10% nonvehicular users.

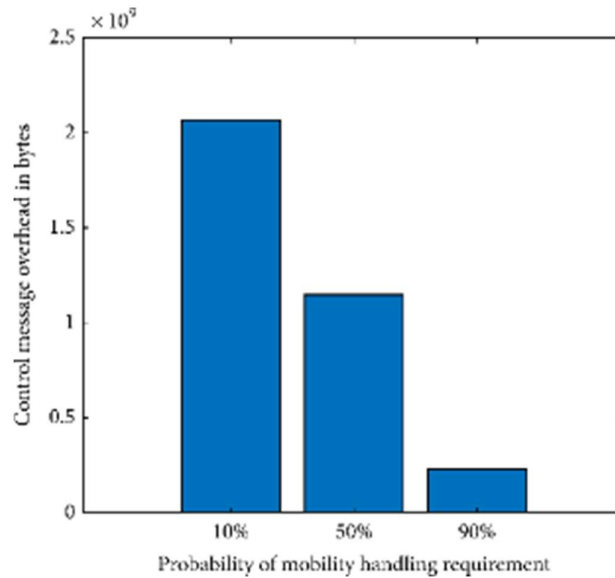


Figure 10 - Control message overhead in case of 50% nonvehicular users.

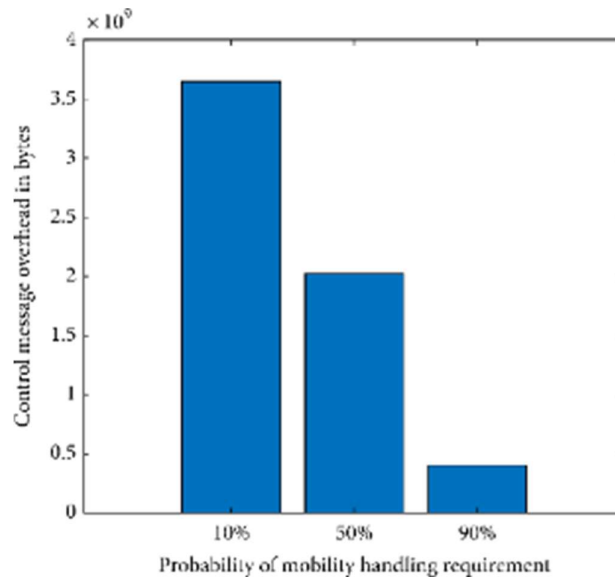


Figure 11 - Control message overhead in case of 90% nonvehicular users.

I reach the most prominent benefit when the ratio of nonvehicular users is high but the number of sessions requiring mobility handling is low. The diagram in Figure 12 shows the overhead is at the 108-byte range, and it increases to the 109-byte range within diagrams from Figure 9 till Figure 11. The more nonvehicular users I have in this scenario, the more time the network has preparing handovers. The fewer mobility-required sessions the users have, the more overhead is caused, as there is no need for mobility handling. Figure 12 summarizes diagrams of Figure 9, Figure 10 and Figure 11 by composing aggregated and compared results.

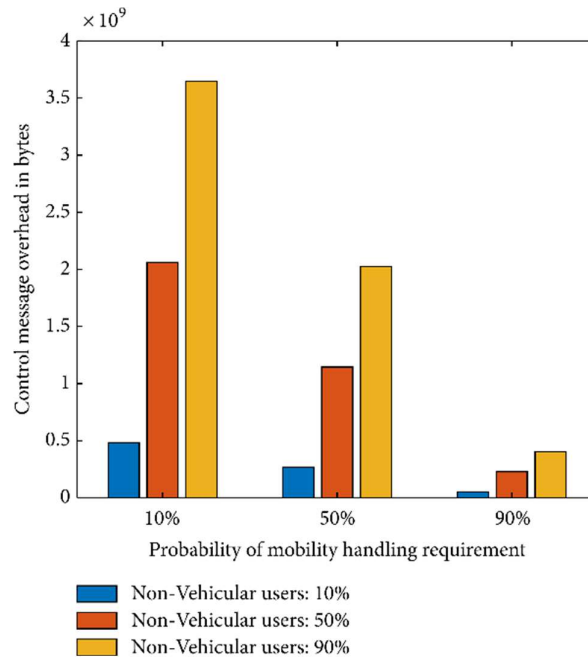


Figure 12 - Summary diagram for control message overhead.

1.4 Thesis Group 2: Cloud-native Mobile IPv6 and Proxy Mobile IPv6

In this thesis group, I will show how MIPv6 HA, PMIPv6 LMA and MAG can look in a cloud-native environment on top of Kubernetes. New features and procedures are enabled by the new design such as autoscaling and automatic failover. I have also identified that, with the help of microservice design, it is possible to assign particular Pods – individual and separated executors - to particular user traffic to booth traffic processing and enable traffic separation. Thus, only those Pods are needed to be scaled out whose traffic has enough high amount. This can save resources in the cloud environment.

1.4.1 Architectural design for Cloud-native Mobile IPv6 and Cloud-native Proxy Mobile IPv6

Thesis 2.1: *I have designed a special Mobile IPv6 Home Agent architecture alternative on the top of Kubernetes called Cloud Native Mobile IPv6 (CN-MIPv6) to meet cloud-native design principles. [C1]. I have also designed special architecture alternatives for Proxy Mobile IPv6 Local Mobility Anchor and Mobile Access Gateway called CN-PMIPv6 on the top of Kubernetes to meet cloud-native design principles [C2].*

The design follows Control and User Plane Separation principle [8] and microservice based design on the top of Kubernetes. Furthermore, it is capable to assign individual executors for particular type of network traffic to boost the processing of Flow Bindings related traffic. Similar design principles are applied for PMIPv6; however, there must be

some new elements: MAG can locally break out the traffic if needed. Furthermore, MAG can be placed to the edge.

1.4.2 Overhead conclusions in a cloud environment for PMIPv6

Thesis 2.2: *I have characterized the containerization overhead specifically for the Proxy Mobile IPv6 control plane. I have designed and implemented a testbed for measuring related performance indicators. I have shown that the used Docker and Kubernetes-based telco cloud environments produce a significant increment (~11%) in average latency, but this is the cost of containerization to utilize its new capabilities. [C3]*

This thesis is about measuring the time between sending a Proxy Binding Update and receiving a Proxy Binding Acknowledgement on MAG and concluding containerization overhead. The detailed results are depicted in Figure 13 and Table 1.

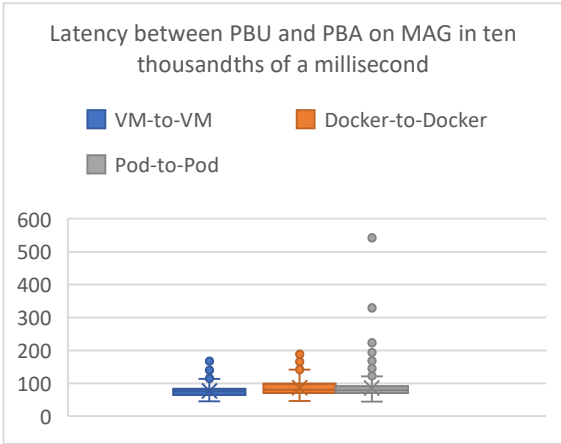


Figure 13 - Box plot of results for the time between sending PBU and receiving PBA on MAG

	Average (ms)	STDEV	MIN	MAX	Median
VM-VM	7.7	1.8	4.5	18.6	7.5
Docker-Docker	8.6	2.4	4.6	20.5	8.0
POD-POD	8.5	2.9	4.4	54.2	7.9

Table 1. - Numerical results of Control Plane measurements

Thesis 2.3: *I have characterized the containerization overhead specifically for the Proxy Mobile IPv6 data plane. I have shown that there is no significant difference between the three evaluated telco cloud environments from the TCP throughput point of view when using IPv6-in-IPv6 tunneling. [C3]*

I have measured TCP throughput in the IPv6inIPv6 tunnel and concluded containerization overhead between LMA and MAG. The detailed results can be seen in Figure 14, Table 2. Note that, the standard deviation of TCP throughput is increased when using Kubernetes Pods.

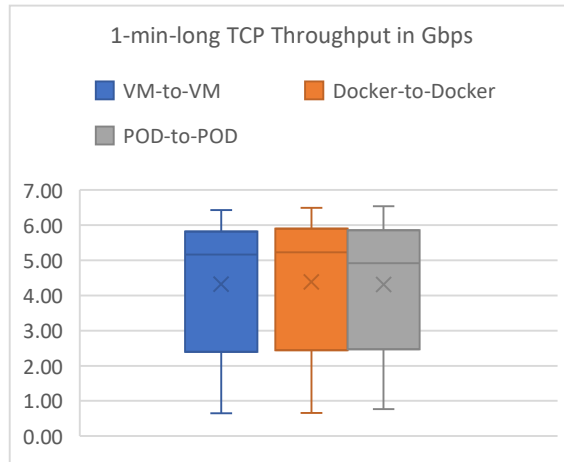


Figure 14 – Box plot for 1-min-long TCP throughput in case of IPv6-in-IPv6 tunneling

	Average (Gbps)	STDEV	MIN	MAX	Median
VM-VM	5.21	0.65	4.15	6.43	5.17
Docker-Docker	5.32	0.66	4.22	6.50	5.23
POD-POD	5.18	0.76	4.17	6.54	4.92

Table 2. - Numerical results of User Plane measurements

1.4.3 Microservice-based design for Cloud-native Mobile IPv6 Home Agent

Thesis 2.4: *I have designed a microservice-based Mobile IPv6 Home Agent architecture on top of Kubernetes. This is the software architecture of CN-MIPv6 Home Agent utilizing Kubernetes-based software components: To analyze the cost of usage of multiple interfaces for a single Kubernetes Pod in the context of the Cloud-native Mobile IPv6 Home Agent control plane, I have designed a model with its several testbed alternatives for evaluating Kubernetes networking connecting the Home Agent entity. I have shown that from the MIPv6 control plane point of view, the usage of multiple interfaces by Multus – as a provider of multiple interfaces for Kubernetes Pods with MACVLAN - increases the jitter (55% on average) but does not have significant latency degradation (2% on average) [C3]*

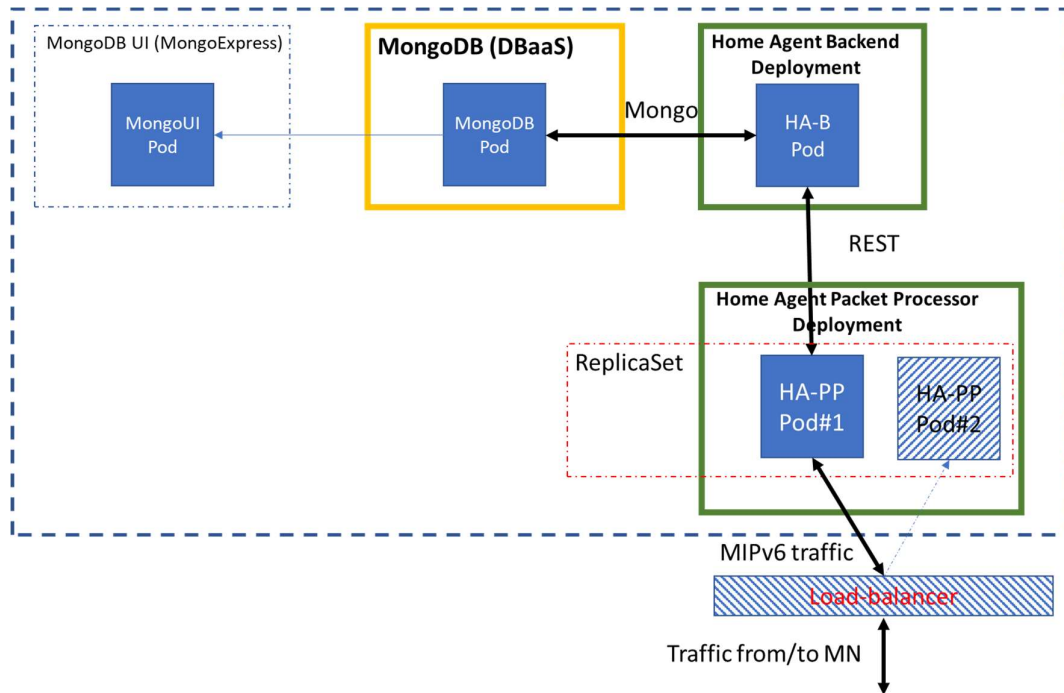


Figure 15 - Microservice design of CN-MIPv6

There are two types of Pods, depicted in Figure 15. 1) The Home Agent Packet Processor (HA-PP) is responsible for processing the incoming MIPv6-related standardized traffic. It communicates via JSON objects with the Home Agent Backend (HA-B). HA-PP is the decision point for allowing MNs' attachment via sending Binding Acknowledgement. HA-B is responsible for communication with external services, in this case, with the BC database (in this PoC, this database is implemented with MongoDB [9]). It is where further additional services connections, e.g., VNF Event Streaming, can be introduced. The goal of HA-B is to hide any outside service communication specialties from HA-PP. Even though there is L7 traffic regarding the overall cloud-native HA, the RFC-standardized L3 interface is as it should be. Database for BC can be either shipped with the Helm charts of cloud-native Home Agent, but it can also be offered in an "as-a-Service" model from the cloud provider. Also, additional Pods can be connected to the database Pod, like Mongoexpress for MongoDB, to monitor BC's content via UI. HA-PP is a separate Kubernetes deployment that allows individual scaling of the number of HA-PP Pods, adjusted to the current needs of packet processing demand. The deployment also allows failover: if an HA-PP Pod fails, Kubernetes will automatically recreate it. Readiness and Liveness Probes have also been introduced for HA-PP, which examines whether the IPv6 Mobility Header Raw socket is open or not. If not, then the Pod is recreated. An external load-balancer can be used to direct the MIPv6-related traffic to the correct Pods. Note that if a failover is executed by Kubernetes deployment, additional configuration may be applied to the system.

Figure 16 depicts the first measurement environment where only Calico is used. The HA-PP Pod has one interface, which is a Calico interface. Not only the MIPv6-related traffic goes over this interface, but Calico internal communication as well (over a logical bridge interface).

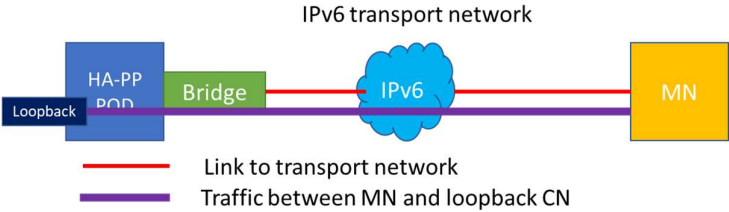


Figure 16 - The testing environment where CNI is Calico

Figure 17 presents the second measurement environment where Calico is extended with Multus.

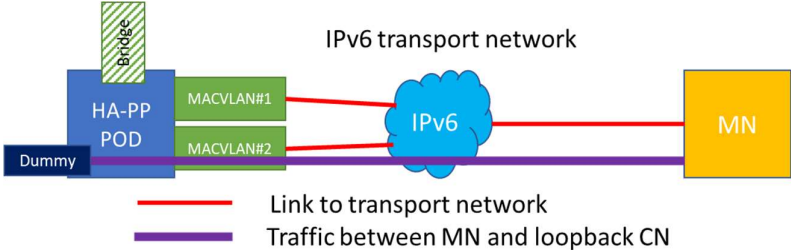


Figure 17 - The testing environment where Calico CNI is extended with Multus

There are multiple MACVLAN interfaces attached. However, the standard Calico interface is there for the internal traffic of Calico (upper bridge interface). In accordance, all the MIPv6-related traffic goes through Multus-created interfaces.

Based on the design mentioned above, I have created a testbed to conclude measurements. Every user-plane measurement is executed between the MN and the corresponding loopback interface of the HA Pod, except the CNI throughput measurement. This measurement has also highlighted the impediments of Calico: its network setup blocks the proper reverse traffic when it is in a different subnet. I needed to circumvent it by setting up a tunnel between the HA and the CN. Overall, the throughput measurement results are acceptable but show a significant feature gap for Calico if applied to a production environment. Measurement in cloud environments could also be biased because the background traffic is unknown. In real-world scenarios, CNs are not part of the OpenStack cluster; they are usually located in other networks. Furthermore, the router is also embedded into the transport network.

By latency, I mean the Round-Trip-Time (RTT) between sending a BU and receiving the corresponding BA from an MN point of view. The measurements have been observed in

both environments. Each CNI trial consists of 10,000 runs (measured by a wrapper bash script around the C-based implementation of MN, timestamps are provided by the Linux date command). Measurement #A presents when the MN newly attaches to a cloud-native Home Agent. Meanwhile, Measurement #B updates an existing BC entry. This provides valuable information on the handover management behavior, as changing the network point of attachment of an already bound MN will mean updating the corresponding BC entry with a new CoA record.

Measurement #A: setting up new bindings

Goal: examine the latency in case of new MN registration to the cloud-native Home Agent.

Conclusion: the latency standard deviation in Multus is significantly higher than Calico's (Figure 18, Table 3). This may increase the attachment time uncertainty of MN.

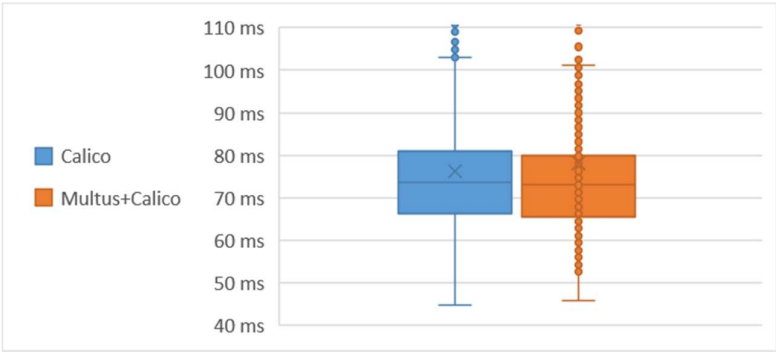


Figure 18 - RTT of new MN attachment (BU and BA sequence while inserting a new HoA)

	AVG	Median	STDEV	MIN	MAX
Calico	76.26	73.7	102.73	44.9	5258.1
Multus+Calico	78.25	73.1	160.32	45.7	5255.2

Table 3. - Numerical results of new MN attachment (ms)

Measurement #B: Updating existing binding

Goal: examine the latency in case of updating an existing entry in the BC database.

Conclusion: for Multus and Calico, the standard deviation of latency is higher than Calico's, which may increase the uncertainty of handover time during mobility events, even though in the case of Multus and Calico usage, the average is a bit lower (Figure 19, Table 4)

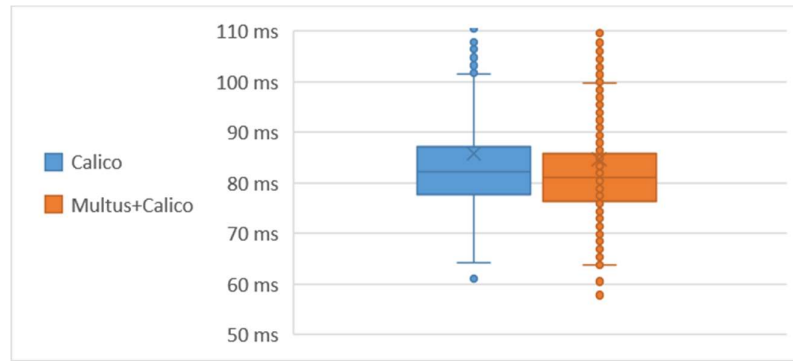


Figure 19 - RTT of updating an existing BC entry

	AVG	Median	STDEV	MIN	MAX
Calico	85.8	82.3	114.15	1.1	215.2
Multus	84.65	81	125.05	7.8	269.7

Table 4. - Numerical results of updating an existing BC entry (ms)

The user plane of CN-MIPv6 relies on IPv6inIPv6 tunneling. The performance of CN-MIPv6 is compared with Calico or Calico + Multus setups from a delay, jitter, and throughput point of view. UDP and TCP traffic are generated by D-ITG [10], except in the throughput scenario where Iperf3[11] is used. Traffic is always originated from MN and terminates at CN. CN is always a loopback interface except for throughput measurements. It is a virtual machine, described in the testbed implementation. Host means the underlying virtual machine acting as Kubernetes Worker node inside the OpenStack cluster.

Measurement #C: Delay and Jitter measurements

Goal: examine the latency and jitter in the case of UDP traffic. Conclude containerization overhead.

Conclusion: from a delay point of view, no significant overhead is added to the containerized HA compared to pure host-based running. This can be seen from the delay difference between the HA and Host measurements, depicted in Figure 20 and Table 5. However, jitter has a remarkable increase. The observations mentioned above are valid for both Calico and Multus-based Kubernetes clusters.

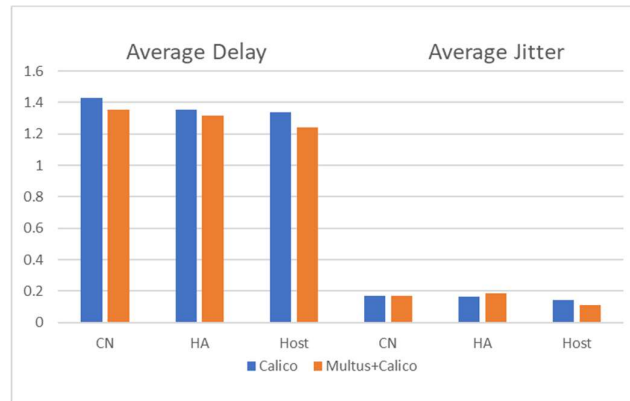


Figure 20 - Average UDP Delay and Jitter Between MN and CN, HA, host

	Receiver	Max Delay	Min Delay	Average Delay	Average Jitter
Calico	CN	9.497	0.946	1.431	0.171
	HA	9.398	0.957	1.352	0.166
	Host	8.538	0.988	1.338	0.144
Mulus	CN	7.846	1.016	1.353	0.171
	HA	7.817	0.957	1.317	0.186
	Host	7.781	0.942	1.24	0.109

Table 5. - Delay and jitter results (in ms) for UDP traffic with Calico and Multus

Measurement #D: TCP Bandwidth measurement

Goal: examine TCP throughput. Conclude containerization overhead.

Conclusion: from the TCP bandwidth point of view, the two solutions give almost identical results (Figure 21); however, in the Multus case, we can add multiple interfaces to increase the bandwidth.



Figure 21 - TCP Bandwidth comparison

Thesis 2.5: I have characterized the cost of usage of multiple interfaces for a single Kubernetes Pod in the context of the Cloud-native Mobile IPv6 Home Agent user plane by measuring the impact of multiple simultaneous IP flow usage. I have shown that Multus - as a provider of multiple interfaces for Kubernetes Pods with MACVLAN - performs significantly better: within the applied parameter set, an average of 52% throughput gain

was presented in the case of TCP. With multiple flow usage, the average jitter is significantly higher in TCP in the case of Multus [C3]

I have shown how TCP throughput performs by utilizing only one (Calico) or two interfaces (Multus) where the CN is an actual VM (not a loopback interface). The expectation is to have significant throughput benefits for Multus as two interfaces can carry traffic separately: one for incoming and one for outgoing from the HA point of view. 10 minutes long TCP traffic was set up between MN and CN nodes using Iperf3. The average throughput of Calico is 271 Mbps. Meanwhile, Multus with multiple interfaces performs with 412 Mbps. Multus with multiple interfaces performs significantly better.

I have also measured the system performance as the number of served flows increases. This is important when multiple applications run on a single MN. Throughput, average delay, and average jitter are examined (Figure 22, Figure 23). Analysing numerous traffic flows may add information to MIPv6 Flow mobility in the cloud environment. Traffic is always traversed between MN and CN (i.e., the loopback interface on HA Pod). Flow bandwidth is 1 Mbps, set by D-ITG, which is 1009 kbps in reality (may slightly differ because of the packet generation method of D-ITG). Packet loss was 0; that's why the relating tables do not contain it separately. The following are the main findings: with multiple flow usage, the average jitter is significantly higher in TCP in the case of Multus: the more TCP flows are used, the higher the average jitter grows. UDP does not show a substantial difference in average jitter and delay points of view. Regarding the throughput, there is no significant difference between the two evaluated CNIs.

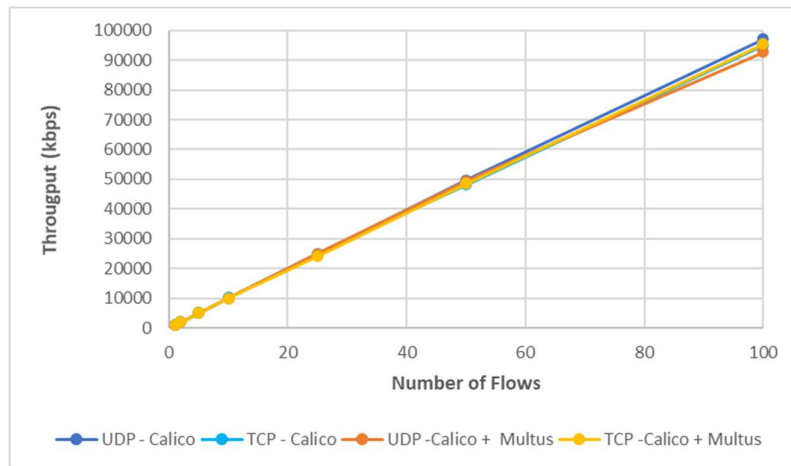


Figure 22 - Throughput compared to the number of flows

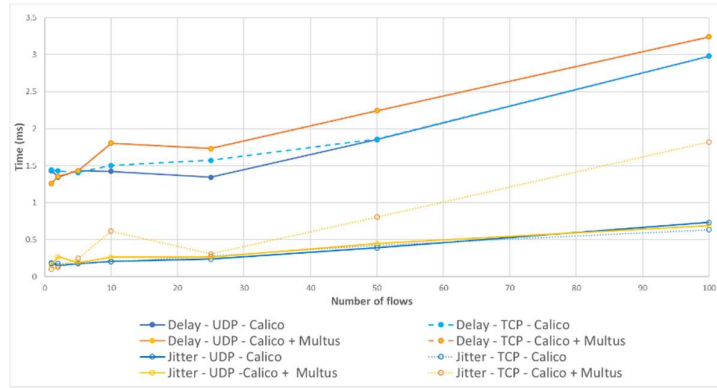


Figure 23 - Average delay and jitter compared to the number of flows

1.5 Closed loop orchestration procedures for Cloud-native Mobile IPv6

1.5.1 Functional and performance evaluation of closed-loop orchestration for Cloud-native Mobile IPv6 Home Agent

Thesis 3.1: *I have proposed a closed-loop orchestration framework for Cloud-native Mobile IPv6 on top of the Open Network Automation Platform. The framework detects the need for failover or scaling for Mobile IPv6 Home Agent instances and can also trigger the execution. [J3]*

Failover is the process when the traffic to a malfunctioning network function is offloaded to a working one. Our measurements logged how long it takes for an automatic failover managed by ONAP toolsets. In the context of CN-MIPv6, this means adding a new HA-PP to a different namespace, and at the end of the failover, this new HA-PP will be the new anchor point.

Scaling is the workflow when additional executors are added or removed to/from the system in order to deal with the changed traffic. In the case of our CN-MIPv6 mobility management service, I emulated how to add a new HA-PP when a certain traffic threshold is reached. At the end of the automatic scaling, both HA-PP components will serve the traffic.

Thesis 3.2: *I have modelled and characterized failover and scaling scenario performances for Cloud-native Mobile IPv6 based on my proposed framework. I have shown that in telco-grade environments, where multiple cloud sites are also available, the failover time can be decreased even to less than 1/30 in average of the commonly*

considered data center MTTR value. Scaling adds 67% more capacity to the system from a TCP throughput point of view. [J3]

The corresponding box plot for failover can be seen in Figure 24. Meanwhile, numerical results are shown in Table 6. The failover is executed within 106.26 sec on average (median: 103.93, stdev: 30.42. The maximum is 210.19 sec, while the minimum is 58.83 sec. During measurements, I experienced that, sometimes, ONAP waited for an uncertain time during the same process. This explains the stdev and the high difference between the minimum and maximum.

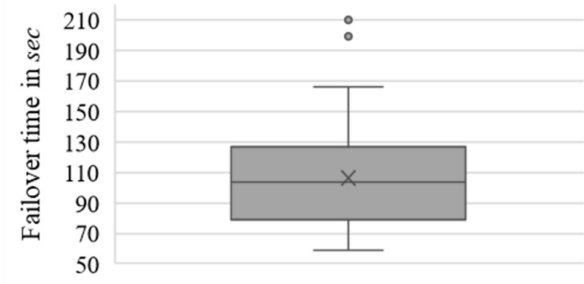


Figure 24 - Failover time results (sec)

MIN	AVG	Median	MAX	STDEV
58.83	106.26	103.92	210.19	30.42

Table 6. - Numerical results of failover time (sec)

In case of scaling, there are two MNs (MN#1, MN#2) which are attached to one HA-PP (HA-PP#1). After the ONAP recognizes the overload of HA-PP#1, then the management system creates a new one. From this point, MN#2 is reregistered via sending BU the newly created HA-PP#2, and the bidirectional tunnel is set up to HA-PP#2 in the case of MN#2. Thus, the CN is reachable via HA-PP#2 in the case of MN#2.

Pods' network interface capacities are limited to 1 Gbps to simulate the limited capacity of a network device and packet processing. This leads to a common base on measurement, and it is easier to compare results. The scaling scenario has been executed 20 times. Figure 25 and Figure 26 show the box plot of the results in Mbps and percentage, respectively. Table 7 shows the numerical results of the average gain for both MNs. After the scaling, the throughput gain for MN#1 is 362.1 Mbps (67.94%), while on MN#2, it is 329.5Mbps (65.4%). MN#2 shows a little less throughput gain, as it has an unconnected period which includes new binding message exchanges and tunnel setup time, while MN#1 is continuously connected to the HA-PP#1.

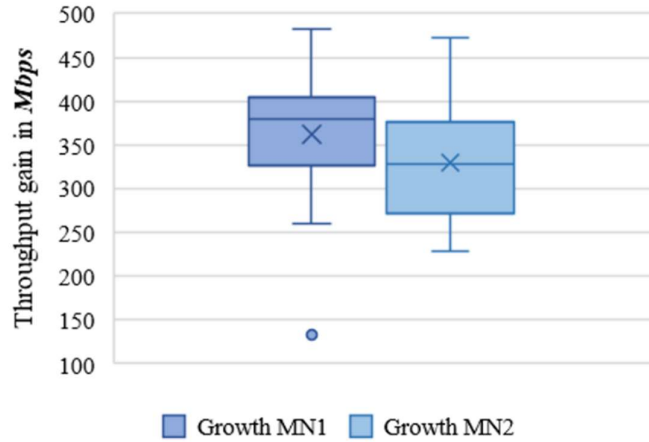


Figure 25 - Throughput gain after scaling out (in Mbps)

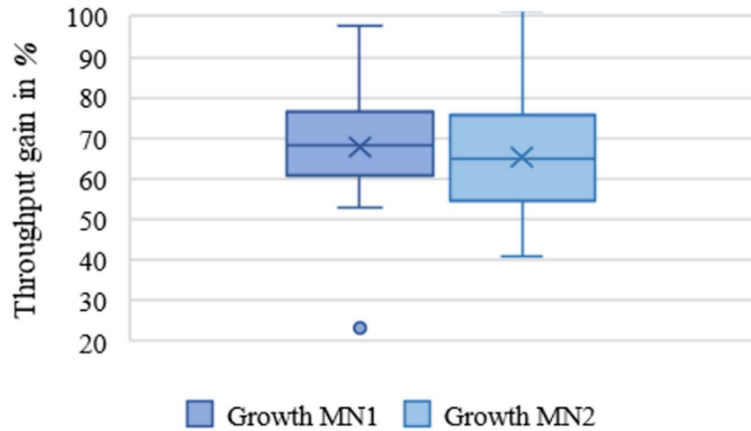


Figure 26 - Throughput gain after scaling out (percentage)

MN1 throughput gain (Mbps)	MN1 throughput gain (%)	MN2 throughput gain (Mbps)	MN2 throughput gain (%)
362.1 Mbps	67.94%	329.5 Mbps	65.4%

Table 7. - Numerical results of average throughput gain after scaling

1.5.2 Numerical calculation

The availability of a telecommunication system is one of the most critical quality metrics. By automating the failover procedure of a network function, the availability of the network function significantly improves.

A short recap, the availability of a single NF can be calculated by

$$(1) A_s = \frac{MTBF}{MTBF + MTTR}$$

where MTBF is Mean Time Between Failures, MTTR is Mean Time to Repair.

Telco services provide high availability; for URLLC services, these expectations are growing further. When the availability expectations for service are in the range of five or six nines for the service components, including the network function software, the availability expectation is even higher. These figures are achieved by applying redundancy schemes. There are no widely agreed/accepted figures for software MTBF and MTTR values, so I calculated with a range of values (MTBF from a month to two years, MTTR from 5 minutes to 45 minutes). I applied different redundancy schemes for these values to ensure that the availability of the redundant solutions is above six nines. Obviously, the reduced MTTR in every case improves the availability. This improvement allows the deployment of lighter redundancy schemes for the network function when its failover is automated and the same availability is still provided. For example, a 2N redundant deployment allows deploying a 3+1 redundant scheme, and the same availability is provided (thus, for four software instances, three instances become active instead of two). Suppose the original MTTR was on the higher end (i.e., above 15 minutes). Even the 6+1 redundant scheme provides the same availability (for all MTBF values) as the 2N redundant solution without failover automation. Thus, a significant amount of resources can be saved. Note that it is also possible to use the "saved availability budget" for other components of the system (e.g., lowering hardware availability by employing less personnel and saving cost) and keep end-to-end availability on the same level or simply offer better availability for customers.

With 2N and 3N redundant systems, the availability calculation is the following:

$$(2) A_{2N} = 1 - (1 - A_s)^2$$

$$(3) A_{3N} = 1 - (1 - A_s)^3$$

Thesis 3.3: *I have calculated the availability of Cloud-native Mobile IPv6 by the results of Thesis 3.2. I have shown that due to the enhanced failover time of CN-MIPv6 in my proposed ONAP-based framework, 2N redundancy can be applied instead of 3N. This means significant cost and resource optimization. [J3]*

In *Thesis 3.2*, I have shown the average time for failover, which is 106.23 sec. This value can be considered as the MTTR of the function I evaluate.

The goal of these calculations is the following: in our ONAP-based failover case, it is possible to reduce the level of redundancy. This means the number of deployed instances can be decreased in order to save resources. Meanwhile, the availability of service is not jeopardized. For simplicity reasons, I apply 99.999% availability for NF, but in a real-world scenario, higher availability is expected (calculation depicted in Figure 27).

1N non-redundant case:

Based on the above-mentioned equations, I calculated the MTBF value for the 1N system if availability must be at least 99.999%, which is 123 days. This means the frequency of system collapse cannot be less than 123 days; otherwise, the system availability cannot reach 99.999%.

I also consider that this MTBF value describes the by-default behavior of our software system.

2N redundant case:

With the MTTR=106.23 sec value and 2N redundancy, the MTBF value is 9.3 hours ($MTBF_{2N}=9.3$ hours) if min 99.999% availability is kept. Thus, if the $MTBF_{2N}$ is 9.3 hours or less, the system is below the target of 99.999% availability. So, if every 9.3 hours, there is an error with the given MTTR value, the system will still operate on at least 99.999% availability.

3N redundant case:

With the MTTR=106.23 sec value and 3N redundancy, the MTBF value is 1.3 hours ($MTBF_{3N}=1.3$ hours) if min 99.999% availability is kept. Thus, if the $MTBF_{3N}$ is 1.3 hours or less, the system is below the target of 99.999% availability. So, if every 1.3 hours, there is an error with the given MTTR value, the system still operates on at least 99.999% availability.

We can save resources if the same availability can be kept with fewer redundant pairs. If we know that our system has an error on average every 123 days, then let's check the max MTTR value that is allowed with 2N redundancy to keep at least 99.999% availability. This is 9.365 hours ($MTTR_{2N}=9.365$ hours). This means that every 2N system with $MTBF = 123$ days can be reduced to our 1N system, where the MTTR is min 9.365 hours. Because this MTTR value reaches a certain level, where the availability of the 2N system does not keep 99.999% even though it may be better than the 1N, but the over-availability requirement is not fulfilled. But using 1N instead of 2N means a service outage as there is no active pair to maintain service. To circumvent this, a 3N redundancy system is needed. A similar logic is applied: if the $MTBF = 123$ days, then let's calculate the min MTTR when the 3N system does not add availability gain (not reaching 99.999%) compared to the 2N system. Every 3N system can be reduced to a 2N system where the MTTR is between 12 min and 65 hours.

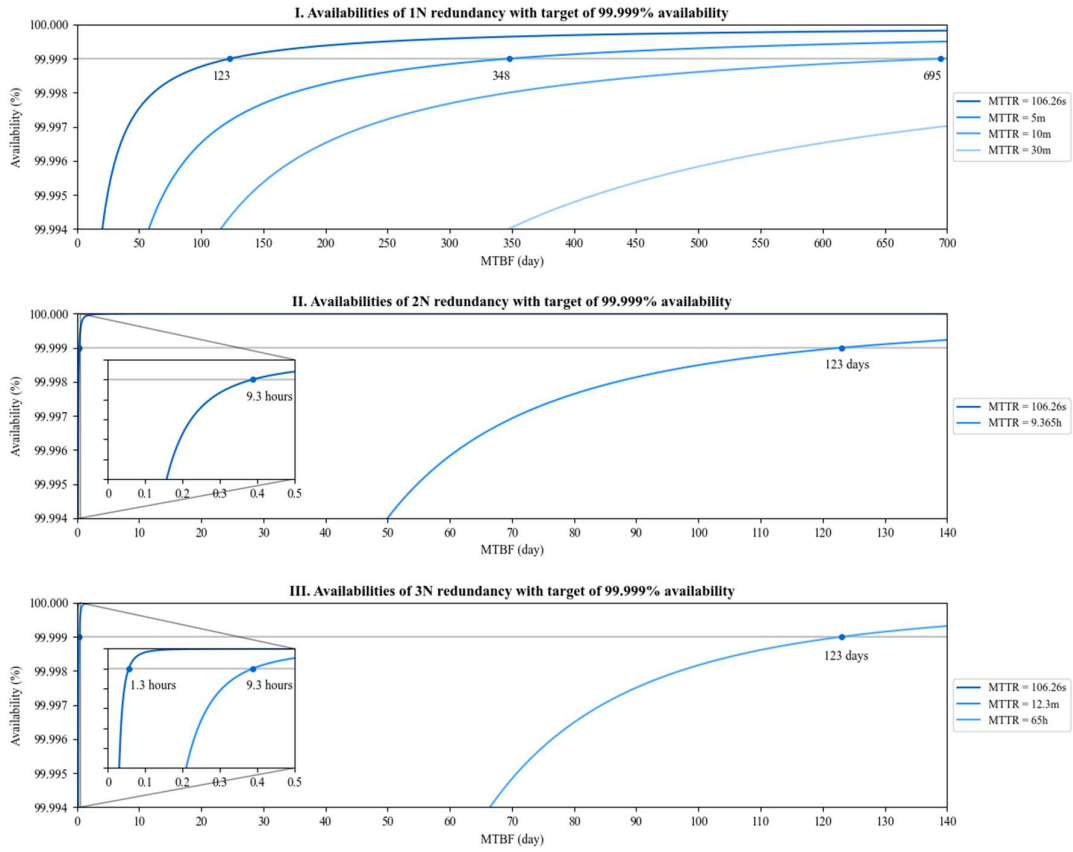


Figure 27 - Redundancy calculation

Application and Conclusion

Traffic offload between 3GPP-based and local (e.g., Wi-Fi) interfaces could save resources on radio frequencies, used by base stations. These frequencies are one of the biggest cost factors of mobile network as they need to be licenced continuously from governmental organisations. This dissertation adds new elements into communication networks to make traffic offload more intelligent as it ensures session continuity and introduces new level of dynamicity. With this application-aware session management, Service Providers can move IP flows between interfaces on-the-fly and users rarely recognise this process.

In a real-life environment, the ownership of the Wi-Fi and the mobile network must be the same or at least they should belong to the same administrative domain. This is an economical and technological limitation of the solution which narrows the scope of target Service Providers. The natural evolution and next step of this work is to prove details economical simulations and calculation to estimate the Capital/Operation Expenditure and financial savings.

As stated in the resource work, several core network elements and sometimes end-user modification are required to utilise the full potential of the solution: User Plane Function, Policy Control Function, Non-3GPP Interworking. These cross-component features are very hard to maintain and develop as they require high-level cooperation between teams and business unites and even companies. Standardisation is a key prerequisite to move forward to any kind of dynamic traffic offload direction, meanwhile several standardisation bodies may be affected e.g.: IETF, IEEE and 3GPP. However, there are several ways, where Wi-Fi networks are attached to mobile networks.

My Publications

[J1] Ákos Leiter, László Bokor, "A Flow-Based and Operator-Centric Dynamic Mobility Management Scheme for Proxy Mobile IPv6", *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 4567317, 21 pages, 2019. <https://doi.org/10.1155/2019/4567317>

[J2] Ákos Leiter, Mohamad Saleh Salah, László Pap, and László Bokor, "Survey on PMIPv6-based Mobility Management Architectures for Software-Defined Networking", *Infocommunications Journal*, Vol. XIV, No 2, June 2022, pp. 2-18., <https://doi.org/10.36244/ICJ.2022.2.1>

[J3] Ákos Leiter, Edina Lami, Attila Hegyi, József Varga and László Bokor, "Closed-loop Orchestration for Cloud-native Mobile IPv6", *Infocommunications Journal*, 2023 1st issue, DOI: 10.36244/ICJ.2023.1.5

[C1] Ákos Leiter, László Bokor, and István Kispál. 2020. An Evolution of Mobile IPv6 to the Cloud. In *Proceedings of the 18th ACM Symposium on Mobility Management and Wireless Access (MobiWac '20)*. Association for Computing Machinery, New York, NY, USA, 137–141. <https://doi.org/10.1145/3416012.3424633>

[C2] Ákos Leiter, Nándor Galambosi, and László Bokor. 2021. An Evolution of Proxy Mobile IPv6 to the Cloud. In *Proceedings of the 19th ACM International Symposium on Mobility Management and Wireless Access (MobiWac '21)*. Association for Computing Machinery, New York, NY, USA, 107–115. <https://doi.org/10.1145/3479241.3486684>

[C3] Á. Leiter et al., "Cloud-native IP-based mobility management: a MIPv6 Home Agent standalone microservice design," *2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, Porto, Portugal, 2022, pp. 252-257, doi: 10.1109/CSNDSP54353.2022.9908059.

[C4] Ákos Leiter, Edina Lami, and László Bokor. 2022. Towards Cross-domain Mobility Management in the Edge: New Elements of Cloud-native Mobile IPv6 Full-scale Implementation. In *Proceedings of the 20th ACM International Symposium on Mobility Management and Wireless Access (MobiWac '22)*. Association for Computing Machinery, New York, NY, USA, 119–122. <https://doi.org/10.1145/3551660.3560919>

[C5] Á. Leiter, P. Böösy, M. Kis, and L. Bokor, 'Performance costs for IPv6-based mobility management on the top of Kubernetes', in *2023 IEEE 9th International Conference on Network Softwarization (NetSoft)*, 2023, pp. 217–221. doi: 10.1109/NetSoft57336.2023.10175456.

[C6] Á. Leiter, A. Hegyi, N. Galambosi, E. Lami, and P. Fazekas, 'Automatic failover of 5G container-based User Plane Function by ONAP closed-loop orchestration', in *NOMS*

2022-2022 IEEE/IFIP Network Operations and Management Symposium, 2022, pp. 1–2. doi: 10.1109/NOMS54207.2022.9789799.

[C7] Á. Leiter et al., ‘Intent-based 5G UPF configuration via Kubernetes Operators in the Edge’, in 2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN), 2022, pp. 186–189. doi: 10.1109/ICUFN55119.2022.9829576.

[C8] Á. Leiter, A. Hegyi, I. Kispál, P. Böösy, N. Galambosi, and G. Z. Tar, ‘GitOps and Kubernetes Operator-based Network Function Configuration’, in NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium, 2023, pp. 1–5. doi: 10.1109/NOMS56928.2023.10154212

[C9] Á. Leiter, "Cloud Stock Exchange for Telecommunication," 2018 International Symposium on Networks, Computers and Communications (ISNCC), Rome, Italy, 2018, pp. 1-4, doi: 10.1109/ISNCC.2018.8531052.

[C10] Á. Leiter and L. Bokor, "Stock market trading strategies and cost estimation for telecommunication resource management," 2019 10th International Conference on Networks of the Future (NoF), Rome, Italy, 2019, pp. 126-129, doi: 10.1109/NoF47743.2019.9015018.

[C11] Á. Leiter and L. Bokor, "A Study on Use Cases and Business Aspects of Cloud Stock Exchange," 2020 International Conference on Information Networking (ICOIN), Barcelona, Spain, 2020, pp. 732-737, doi: 10.1109/ICOIN48656.2020.9016536.

References

- [1] C. Perkins (Ed.), D. Johnson, and J. Arkko, ‘Mobility Support in IPv6’. in Internet Request for Comments, no. 6275. RFC Editor, Fremont, CA, USA, Jul. 2011. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc6275.txt>
- [2] S. Gundavelli (Ed.), K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, *Proxy Mobile IPv6*. in Internet Request for Comments, no. 5213. Fremont, CA, USA: RFC Editor, 2008. doi: 10.17487/RFC5213.
- [3] G. Tsirtsis, H. Soliman, N. Montavont, G. Giarretta, and K. Kuladinithi, *Flow Bindings in Mobile IPv6 and Network Mobility (NEMO) Basic Support*. in Request for Comments, no. 6089. IETF, 2011. [Online]. Available: <http://www.ietf.org/rfc/rfc6089.txt>
- [4] C. Bernardos (Ed.), ‘Proxy Mobile IPv6 Extensions to Support Flow Mobility’. in Internet Request for Comments, no. 7864. RFC Editor, Fremont, CA, USA, May 2016. [Online]. Available: <https://www.rfc-editor.org/rfc/rfc7864.txt>
- [5] H. Zhang, W. Ma, W. Li, W. Zheng, X. Wen, and C. Jiang, ‘Signalling Cost Evaluation of Handover Management Schemes in LTE-Advanced Femtocell’, in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*, May 2011, pp. 1–5. doi: 10.1109/VETECS.2011.5956481.
- [6] ‘Open Network Automation Platform (ONAP)’. Accessed: Nov. 05, 2023. [Online]. Available: <https://www.onap.org/>
- [7] J. Varga, A. Hilt, C. Rotter, and G. Járó, ‘Providing Ultra-Reliable Low Latency Services for 5G with Unattended Datacenters’, in *2018 11th International Symposium on Communication Systems, Networks Digital Signal Processing (CSNDSP)*, 2018, pp. 1–4. doi: 10.1109/CSNDSP.2018.8471756.
- [8] ‘3GPP TS.23.314 - Architecture enhancements for control and user plane separation of EPC nodes’. Accessed: Jan. 22, 2024. [Online]. Available: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3077>
- [9] ‘Mongodb’. Accessed: Nov. 05, 2023. [Online]. Available: <https://www.mongodb.com/>
- [10] S. Avallone, S. Guadagno, D. Emma, A. Pescape, and G. Ventre, ‘D-ITG distributed Internet traffic generator’, in *First International Conference on the Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings.*, 2004, pp. 316–317. doi: 10.1109/QEST.2004.1348045.
- [11] *iPerf - The network bandwidth measurement tool*. [Online]. Available: <https://iperf.fr>