Enabling the Management and Orchestration of Virtual Networking Functions on the Edge

Vincent Melval Richards[1][a], Rodrigo Moreira[1,2][b] and Flávio de Oliveira Silva[1][c]
[1]*Faculty of Computing (FACOM), Federal University of Uberlândia (UFU), Uberlândia/MG, Brazil*
[2]*Institute of Exact and Technological Sciences, Federal University of Viçosa (UFV), Rio Paranaíba/MG, Brazil*

Keywords: Cloud Computing, Edge Computing, NFV, Virtualization, IoT.

Abstract: Tthe Internet of Things (IoT) is the pillar of the next generation of applications that will create a fusion between the real and digital worlds. Several use cases rely on Edge computing capabilities, and the virtualization capacity on Edge is a crucial requirement. The management and orchestration of computing, storage, and networking resources in the Edge cloud pose several challenges. In this work, we propose a Management and Orchestration (MANO) of Virtual Networking Functions (VNFs) on the Edge using standard solutions such as Open Source MANO (OSM) and OpenStack. To showcase our approach, we handled an experimental evaluation using an RPi3 infrastructure. OpenStack manages this infrastructure, and OSM provides the VNF MANO capabilities offering a significant improvement to support cloud computing on Edge. The evaluation showed the feasibility of using low-cost devices such as RPi with standard management solutions used in the core cloud.

1 INTRODUCTION

The growth and popularity of IoT pull along with its related areas such as cloud computing, machine learning, and big data. The noted long term challenge of how to load the vast amount of data generated by long-term accumulation of streaming data or video data (such as Augmented Reality (AR), Virtual Reality (VR), and Mixed Reality (MR)) with IoT hardware devices (Kristiani et al., 2019) could be address by the sharing of larger platform with IoT platform as one form of solution.

Cloud Computing is of vital importance to businesses and individuals and is serving its purpose well (Zhang and Ravishankar, 2019). However, the cloud data centers are far away, and the distance that data need to traverse the Internet to end-users is also fairly long, here arises the problem of latency (Zhang et al., 2018). Numerous proposals have been established, such as (Gouareb et al., 2018), that aim to mitigate the effect of structural latency, through optimization techniques, that the conventional service implementation model imposes.

[a] https://orcid.org/0000-0003-2850-9602
[b] https://orcid.org/0000-0002-9328-8618
[c] https://orcid.org/0000-0001-7051-7396

Latency is acceptable for some applications that can tolerate delays; however, an application such as Virtual Reality, Real-time Processing, and Augmented Reality will function poorly or even fail when faced with latency issues (Shi et al., 2016).

Furthermore, the advent of the 5G network presents another pressing need to address the latency challenges that Cloud Computing faces. Therefore Edge Computing is presented as a solution to resolving the latency challenges. Edge computing can achieve this by bring processing closer to end-users at the Edge of the network and also allow the continuation of processing if the internet connection is disrupted (Hu et al., 2015).

Cloudlet was put forward as a means to implement MEC with OpenStack++, which is a lighter version of OpenStack, but according to the researchers (van Kempen et al., 2017) OpenStack++ platform was meant to execute on powerful server machines, which would make it challenging to deploy in the realistic geo-distributed setting. However, in our work, we were able to install OpenStack++ on RPi on Ubuntu 16.04 *armhf* in a KVM virtual machine. Our new approach was to install OpenStack nova-compute on Ubuntu 16.04 server installed on the RPi3, and the results prove to be more satisfying.

Our contribution to the process of enabling virtu-

Richards, V., Moreira, R. and Silva, F.
Enabling the Management and Orchestration of Virtual Networking Functions on the Edge.
DOI: 10.5220/0009398203380346
In *Proceedings of the 10th International Conference on Cloud Computing and Services Science (CLOSER 2020)*, pages 338-346
ISBN: 978-989-758-424-4

alization at the edge of the network for future service provision includes the following:

1. Provide another alternative to the available options currently being used at the edge of the network on resource constraint devices such as the raspberry pi.
2. Demonstrate the feasibility and possibility of utilizing RPi at the edge as a low-cost and low energy consumption device. Therefore aid in hastening the development of Mobile Edge Computing (MEC) and implementation of 5G network. Our work, along with related research, could serve as a motivation for other researchers and investors to get involved in the development process of Edge Computing, which is a facilitator for the implementation of MEC and 5G computing.
3. Introduce an ETSI-compliant infrastructure for service deployment at the edge.

The remaining of the paper is organized as follows: In Section 2, we place our work in the context of related work concerning management and orchestration of Virtual Networking Functions (VNF) on the Edge. In Section 3, we describe our solution and its features to cope with state-of-the-art challenges. Section 4 describes a proof-of-concept scenario and the evaluation methodology to asses our solution feasibility. Section 5 presents the measurements and a discussion about them. Finally, in Section 6, we summarize our findings, and we point out some research directions.

2 RELATED WORK

Edge computing research and development have reached a high point of interest. It will be the facilitator of 5G and Mobile Edge Computing as it seeks to bring processing closer to the user at the Edge of the network and, in so doing, help to resolve the latency problem. However, there still exist hurdles to get over, and the literature describes several works related to this aim.

The authors of MEC-ConPaaS (van Kempen et al., 2017) project stated that deploying a realistic mobile Edge cloud remains a challenge because mobile operators have not yet implemented MEC technologies in production, and this makes it impossible for researchers to use their techniques in actual mobile phone networks.

They also noted that there exist very few open-source platforms that may support experimentation that emulate a mobile Edge cloud, and they recognize the most mature one to be OpenStack++ (van Kempen et al., 2017).

However, MEC-ConPaas researchers claimed that the general OpenStack implementation relies on classical server machines for its deployment. Their proposal is to deploy a experimental MEC testbed that relies on single-board computers such as Raspberry Pis (RPis) and similar devices. Their work relies on ConPaaS (Pierre and Stratan, 2012), which is an open-source run-time management system for elastic applications in public and private cloud environments. They utilized RPi with Linux Containers (LXC) enabled and OpenStack orchestrating the containers.

Therefore they proposed a more natural way of deploying an experimental MEC testbed that relies on single-board computers such as Raspberry Pis (RPis) and similar devices. Their work relies on ConPaaS (Pierre and Stratan, 2012), which is an open-source run-time management system for elastic applications in public and private cloud environments.

MEC-ConPaaS presented an interesting foundation and proof of the concept of using the raspberry pi at the edge, which was further highlighted by (Pahl et al., 2016) who did similar research with ConPaas and the implementation of LXC with three hundred RPis in a cluster.

Also, they explored the various forms of a Platform-as-a-Service (PaaS) for edge computing. However, these relevant foundation researches may be impacted because of the shift from the use of LXC to newer technologies such as Kubernetes and LXD, along with security concerns that it presents. Therefore a newer container technology called LXD based on LXC was presented by (Ahmad et al., 2017) who proposed the use of RPi2 Model B and LXD Linux containers.

(Kristiani et al., 2019) implements Edge Computing Using OpenStack and Kubernetes. Their implementation uses three layers, such as the Cloud side, Edge side, and Device side. The cloud side mainly deals with more complicated operations and data backup. Its main function is to deploy the Kubernetes cluster on an OpenStack platform. Also, the cloud side contains the Virtual Machine (VM) and Kubernetes master side, and it also deals with data backup, complex operations, data visualization, and other applications that do not need quick responses.

A more sophisticated form of edge implementation from the previously mention containers was introduced by (Chang et al., 2014), which applied a computational model called the Edge Cloud. The application here brings the cloud to the edge and uses it as a means of enabling a new type of hybrid application called Edge Apps, which runs over Openstack

(Chang et al., 2014).

Another edge app implementation was presented by (Schiller et al., 2018), who creates mobile edge apps that are managed as virtual network functions. Their platform also includes an SDN controller to manage traffic by using the control plane to derive states for traffic management. They research Vehicular Fog Computing for Video Crowd Sourcing, where they analyze the availability of vehicular fog nodes based on a real-world traffic dataset. Then explore the serviceability of vehicular fog nodes by evaluating the networking performance of fog-enabled video crowdsourcing over two mainstream access technologies, DSRC and LTE (Zhu et al., 2018).

(Bellavista et al., 2018) implement a solution that uses powerful and low-cost middleboxes deployed at the edges of the network. It serves as enablers for their Human-driven Edge Computing (HEC) as a new model to ease the provisioning and to extend the coverage of traditional MEC solutions. Their approach uses different devices from the RPi but is also relevant as middleboxes allow specialized services to be used at the edge of the network. However, some of these services can be enabled on the RPi to reduce cost of implementation.

(Tong et al., 2016) apply hierarchical architecture for the edge cloud to enable aggregation of the peak loads across different tiers of cloud servers. They use this as a means of maximizing the number of mobile workloads being served. The need for load balancing is essential and can also be implemented on the RPi which have multipurpose capabilities which are similar in some aspect to the PC.

Another load balancing technique is proposed by (Puthal et al., 2018) whose technique is used to authenticate the EDCs and to discover less loaded Edge Data Centers (EDC) for task allocation. They state that it is more efficient than other existing approaches in finding less loaded EDC for task allocation and that it strengthens security by authenticating the destination EDCs.

In the work (Lei et al., 2018), the authors' solution allows mobile Edge applications to be provided by chaining the service functions with the assistance of Edge computing techniques and virtualized resources. They also build a testbed to evaluate their Edge caching architecture for proof of concept and tested it with typical caching scenarios. The research is not base on the RPi; however, the chaining of function can be tested on the RPi.

In conclusion of the review of related work, the use of the RPi with other devices is assessed in (Akrivopoulos et al., 2018) who uses RPIs and Zotac and also an off-the-shelf Intel-based edge-based server box. The software that they deployed on the devices is bundled using a collection of Docker containers. They use these to create an IoT-based platform for real-time monitoring and management of educational buildings on a national scale. They design the system to process sensor data on the Edge devices of the network.

3 MULTI-LAYERED EDGE-CORE VIRTUALIZATION APPROACH

This section presents the rationale behind the multi-Layered Edge-Core virtualization approach and the main building blocks of the architecture designed.

3.1 Solution Rationale

To address the challenge of uniform infrastructure and application life-cycle management, we propose a jointly Edge-Core infrastructure management approach. According to Figure 1, the proposed solution deals with two compute infrastructure technologies. First, the bare-metal witch comprises the traditional data centers that offer public and private cloud services on top of commodity hardware. Second, it concerns low-cost computing where infrastructure resources could be in strategic locations, preferably close to the user. These facilities could place in malls, sports stadiums, homes, airports, and parks. The state-of-the-art proposals do not address the hybrid aspect of service deployment and management. Although, they do not handle uniformly, heavy virtualization on top of the low-cost and high-capacity hardware simultaneously.

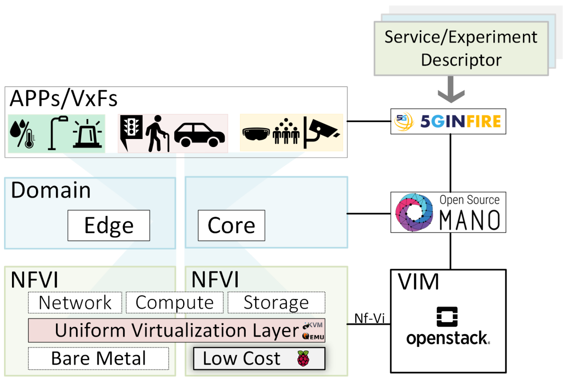

Figure 1: Conceptual Architecture.

Our solution is layered and maintains features close to the European Telecommunications Standards Institute (ETSI) NFV framework. Considering a bottom-up approach, we describe the Network Functions Virtualization Infrastructure (NVFI), the hardware management entity, as a biform infrastructure layer, so the virtualization engine is alike for low-cost, high-performance servers. Thus, it becomes possible to offer Infrastructure as a Service (IaaS) in which the service is uniformly deployed over both hardware using the same manifest file (descriptor). Thereby, we make the ability to deploy services across multiple domains hybrid. Also, next to the NFVI layer comprises the Virtual Infrastructure Manager (VIM), the layer that controls and manages the infrastructure entities.

On top of the bottom layer, we propose an intermediate tier that comprises the service deployment domains. It contains the Core and Edge blocks, which are respectively infrastructures that host services traditionally on high-performance hardware and low-cost infrastructures that can be deployed and enabling verticals of services such as IoT, Smart farms, and others. MANO compliant solutions could be placed here to handle the service management and orchestration of the middle tier. State-of-the-art solutions deal with these deployment domains separately.

The topmost layer that makes up the framework of the proposed solution comprises the multi-form applications that we refer to as Virtualized Everything Functions (VxFs), according to (Silva et al., 2019). Our proposal as a uniform virtualization layer for both bare-metal and low-cost devices democratizes the deployment domain, so the plethora of applications are broader than state-of-the-art solutions. Also, testbeds of applications like (Silva et al., 2019), (Sallent et al., 2012), and (Silva et al., 2018) can add to their facilities our infrastructure model that can add functionality as service descriptor manifest files.

Virtualization is essential for Edge computing provisioning and enables in most use-cases for new kinds of applications (Paolino et al., 2015). Therefore, any device that will be used on the Edge must be able to accommodate virtualization even if it is operating system level, such as Linux Containers (LXC) or any other container technology.

However, the use of a virtual machine is needed for some applications in cases where better security, isolation, and network performance (d. S. Marques et al., 2018) are preferred, but on most small, low-cost devices enabling or using virtualization is more difficult, especially if one wants to use virtual machines (Barik et al., 2016). The use of virtual machines on the RPi is relatively new and is only enabled through the use of particular operating systems such as OpenSuse, Ubuntu, and Gentoo.

In order to build an Edge-Core virtualization platform to host VxFs spanning in different use-case domains, we have separated the architecture into two verticals to make it loosely coupled. The state-of-the-art solutions deal with VxF deployment specifically for each domain, and each virtualization technology: low-cost and high-performance virtualization requires specific managers (Li et al., 2017). For instance, OpenNebula (Edge release) (Milojičić et al., 2011) deploy services on low-cost hardware; however, uniformity on it does not prevail. That is, services as conceived as LXC on low-cost device services, and for bare-metal follows standard virtualization.

Unlike other Edge virtualization proposals, our solution enables VNF life-cycle management smoothly. VIM does not need to handle low-cost compute infrastructure or bare-metal separately. Somewhat we adapted a uniform virtualization layer for both types of hardware. Thus, the service deployment on top of the low-cost computing does not necessarily have to be container-based once those solutions have network connectivity drawbacks.

3.2 Architectural Design

To handle these drawbacks we bring a uniform solution as described in Figure 2. Our proposal enables *x*86_64 virtualization compatibility on top of low-cost compute resources as it is straightforward on bare-metal compute resources. The boot process, which starts on Nova-API (release 13.1.4), receives the requests to a new VxF creation. Afterward, the messages go through the RabbitMQ mechanism to the Nova-Scheduler, which decides which *compute-node* contained in the cluster (Cellv2) will handle the request. Ultimately, the remaining build-block components are Keystone 9.3.0, Glance 12.0.0, and Neutron 8.4.0.

The filter (*ComputeCapabilitiesFilter*) in Nova-Scheduler checks if the deployment request fits adequately on the low-cost *compute-node*, according to VxF flavor, being possible the nova-boot process will create and instantiate the VxF. If a low-cost *compute-node* is unable to handle the deployment request, the VxF will be launched on top of bare-metal compute resources.

Commonly RPi has only one physical network interface, and the basic OpenStack deployment requires two interfaces, we proposed an approach based on Open vSwitch (OvS). Therefore, we created two virtual interfaces *vnet0* and *vnet1*, which we assign different addressing plane. Both interfaces were associated with a *OvS*, which had as trunk port the only physical interface of RPi. The RPi physical interface connects directly to a physical switch where

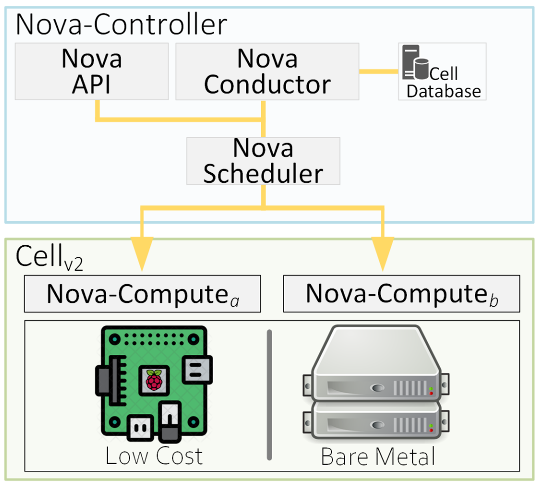

Figure 2: Proposed Solution Build-Blocks.

the controller-node and the network gateway are also connected. Thus, one virtual interface served as a provider interface for the Neutron plugin, working as L2 Agent, and the other allowed the compute-node to exchange messages with the controller-node services such as AMQP and databases.

Our solution for enabling Edge computing involves the use of the small and low-cost RPi running Ubuntu 16 *armhf*-capable. However, its limited memory, in comparison to the bare-metal poses restrictions on the programs that can be installed and executed on it. Therefore, since we are installing OpenStack on the RPi, only the Nova-Compute can be installed on it and not the Nova-API or Nova-Cert as these freezes the RPi as soon as they start running. However, we build it with Nova-Compute and Neutron for the network. Therefore, the RPi memory capacity was able to allow these to run smoothly.

As long as OpenStack requires virtualization, we enable such technology for the Edge with this approach. To this end, we implement OpenStack Nova-Compute to utilize virtual machine by connecting a localized OpenStack controller to our RPi OpenStack *compute-node*. The RPi configuration enables the booting, stopping, accessing, and deleting virtual machines from the RPi.

Therefore our solution can place all these added functionalities at the Edge of the network closer to the user. The use of low-cost devices will also benefit the users and businesses, along with the benefit of low latency will be achieved when the services and functionality are placed at the Edge of the network closer to the user.

Our proposal investigates use of standard virtual machines on low-cost hardware. The goal is to use the same virtual machine that usually runs on high-performance hardware in order to verify the suitability and compatibility of the use of a single MANO entity, in our case OSM. Container-based solutions are known (von Leon. et al., 2018) on these scenarios; however, the suitability of a uniform proposal for deploying VxFs according to the ETSI framework was not discussed in previous studies.

4 EXPERIMENTAL SETUP

To evaluate our solution, we propose an experimental scenario in a real testbed environment. This section presents the testbed that realizes the proposed solution, the experiment and the method used for its evaluation.

4.1 Testbed Description

The architecture of our experiment includes a RPi3 (Nova-Compute$_a$) and a bare-metal desktop computer ((Nova-Compute$_b$) which are together in the network edge. The bare-metal desktop can run services deployed in edge cloud while the RPi3 is the edge computing near the user. Besides that, there is an OpenStack Controller, as presented in Figure 2.

The RPi3 is a Model B+ with a Broadcom BCM2837B0 system on a chip (SoC) with a Cortex-A53 (ARMv8) 64-bit 1.4 GHz processor and 1 gigabyte of LPDDR2 SDRAM. The bare-metal computer and Controller are Core I7 (i7-8550U) processors with 8 gigabytes of RAM. The RPi3 has Ubuntu 16.04 *armhf* server, customized to ARM processors, and publicly available. The software that was installed and configured on the RPi include neutron and nova-compute.

The bare-metal desktop computer uses a Ubuntu 16.04 64-bit (AMD64) server image. It also has the neutron and nova-compute OpenStack components (Mitaka release).

The controller computer uses a Ubuntu 14.04 64-bit (AMD64) server image and several OpenStack components such as Keystone, Glance, Nova, Placement, Neutron network, and Manila.

All the hardware was interconnected using a local physical network. A bridge network configuration supported the connection of the virtual machines created inside the RPi3, the local network.

The overall setup of the experiment is presented in Figure 3. A user uploads a Network Service Descriptor (NSD) to OSM. This service descriptor has an indication to create Virtual Machines (VMs) on the RPi3 located in the Edge. OSM interacts with the

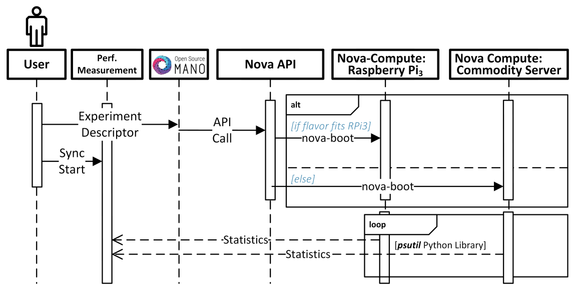

Figure 3: Experimental Scenario Sequence Diagram.

Virtual Infrastructure Manager (VIM), which is represented by the controller computer described above and creates the VMs during the experiment using the nova-compute service running in the RPi3. The setup also enables the creation of VMs in the commodity server represented by the bare-metal desktop computer with is the edge cloud compute server.

4.2 Evaluation Methodology

To find out the capability of the RPi3 to support traditional VMs, our experiments look standard system resources usage parameters such as CPU, RAM, and disk swap usage. Also, we investigate the processor temperature.

The percentage of CPU performance is one such metric that was collected for analysis. Memory response was also necessary as it provides relevant information regarding the amount and types of programs that the system will be able to accommodate. The use of swap was also necessary to buffer RAM after all the system memory has been allocated.

The temperature, which is also related and relevant to the CPU performance, was also recorded for analyzed. Finally, the virtual machines boot times were collected to compare the boot time when only one virtual machine is running as opposed to progressively running multiple virtual machines simultaneously.

During the experiments, we used the *psutil* tool (Rodola, 2016) to collect the data about resource usage on the RPis. We executed each test 20 times to avoid statistical bias and calculated average values.

5 RESULTS AND DISCUSSION

Using the experimental setup and following the evaluation methodology presented in Section 4 the solution presented in this work was evaluated.

The following charts present the results of our experiments. We repeated the experiments to collect the boot times, fifteen rounds.

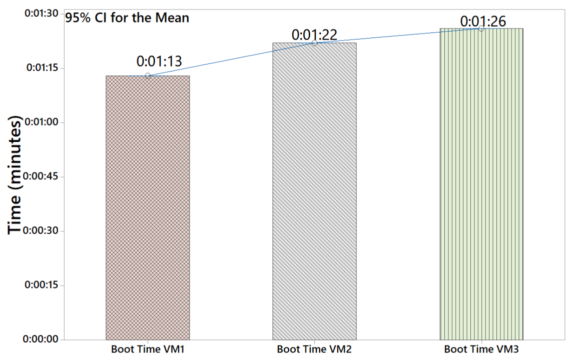

Figure 4: Instances Total Boot Times.

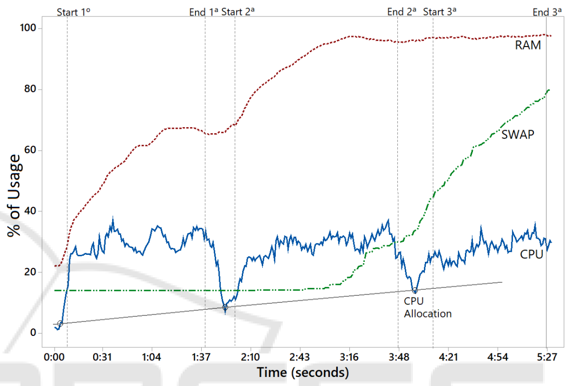

Figure 5: Resource Allocation over Time.

Figure 4 displays the boot times in minutes. The time present is the average time. In each round, we created three VMs since the RPi3 did not support more than this number of instances.

Figure 5 presents the % of the usage of CPU, disk swap, and RAM combined in one chart to give an overall view of the system performance. This view makes it possible to compare and assess these related system performance metrics quickly. The graph shows the evolution of each of the above parameters during the experiment. The system resource usage starts the measurement at the beginning of the experiment. It continues getting this information while the three different VM instances are created. The values presented in the charts is the average of the fifteen rounds of the experiment.

After analyzing the virtual machines booting times, it can be seen from Figure 4 that the booting times of each new virtual machine continue to increase for each new virtual machine that is booted. Therefore after the first virtual machine is booted on the RPi3, the boot time for each subsequent virtual machine will increase; however, it will only increase by less than one minute.

The analysis of the CPU performance during the virtual machines booting process per rounds shows that at the start of the booting process of a virtual machine, the base value of the CPU is low then rises sharply and remains high while fluctuating. Then again at the start of the second virtual machine on the RPi, the CPU value falls sharply first then rises again sharply however the base of the low CPU value now shows an increase. This falling and rising behavior during the virtual machine booting process continues with the booting of the third virtual machine along with the low base value rising again significantly.

Also, we depict in Figure 6 the temperature rise per experiment round. As we can see, it is possible to notice the increase as the instances are launched. The average temperature, considering the 95% confidence interval, is ≈ 55.29°C.

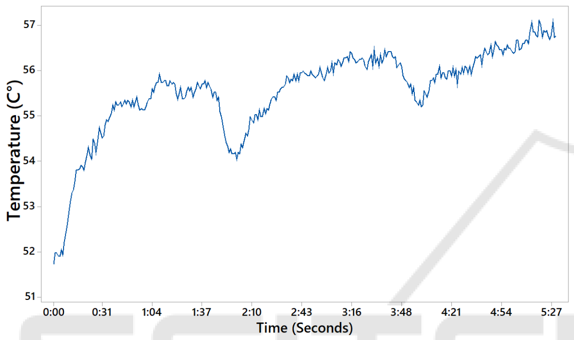

Figure 6: Temperature Measurement.

Also, the analysis of the temperature provided by Figure 6 shows that it rises during the process of booting each new virtual machine. However, after the booting process of each virtual machine, it falls significantly but not very low. Then risings steadily but sharply.

After the creation of a new virtual machine, the RAM allocation, and consumption increases. The memory capacity is the major restricting element in the number of virtual machines that the RPi3 can support. The RPi3 used in the experiment has one gigabyte of RAM. Considering that Ubuntu 16.04 *armhf* image requires 256 RAM per virtual machine, the current limit is three virtual machines.

However, when RAM consumption is near its limit, the swap memory is used as a buffer for processing to prevent the system from crashing. In Figure 5 swap can be observed to be steady before and after one virtual machine is created. Then starts to increase with the creation of the second virtual machine until it peaks with the creation of the third virtual machine.

It can be be seen that the CPU, RAM, swap, and temperature are connected to an extent, and all are influenced by the creating of the virtual machines, especially the last two. All the parameters show an increase during the last two boots, especially with the booting of the third virtual machine.

The MEC-ConPaaS project (van Kempen et al., 2017) states that they use LXC on the nova-compute, which is an RPi also, they state that their first LXC launch image test took 10 minutes to launch them with many configurations they were able to get it to 90 seconds. The boot time of an instance on our Edge infrastructure average less than 2 minutes, which is relatively close to the 90 seconds booting of instances in LXC, which are not as secure as virtual machines.

Finally, we were able to launch images from a MANO miles away. Here we connect the controller to OSM, and this allows us to also start virtual machines on the Edge with virtual network functions as needed or requested. Using this approach then makes it possible to monitor and manage virtual machines on the Edge much more straightforward, using standard platforms common used.

6 CONCLUDING REMARKS

Within this work, we showcased a unique approach to manage and orchestrate resources on servers on the core and low-cost hardware on edge using the same MANO entity, in this case, represented by OSM. We also could explore the constraints associated with the use of the RPi on edge regarding standard parameters associated execution of virtual machines on these devices.

Previous approaches to deal with state-of-the-art OpenStack implementation on the edge such as OpenStack++ (Ha and Satyanarayanan, 2015), and MEC-ConPaaS (van Kempen et al., 2017) were initiated to encourage faster Edge computing development with cloudlets. However, OpenStack++ was meant for powerful servers and was not made with consideration of low-cost devices that are prerequisites for widespread adaptation of edge computing.

The reason for this is that the RPi is a low-cost and low power consumption device (Bekaroo and Santokhee, 2016). Therefore this article shows that the RPi can be used for virtualization on the Edge, and more importantly, it demonstrates that virtualization can be enabled with OpenStack using traditional instances on RPi. Furthermore, it has another advantage where the controller resources can be shared with the RPi when a virtual machine is booting from the RPi. This allows the launching of more significant instances and more memory-hungry virtual machines to meet the needs of applications that need more memory than the RPi limited memory will allow.

According to performance tests, memory is the resource that most degraded as low-cost hardware served virtual machines. This paves the way for new memory allocation formats on low-cost devices.

Our infrastructure proposal based on the ETSI framework made it possible to launch virtual machines uniformly. Namely, MANO does not need to deal with different virtualization technologies for each hardware (high-performance and low-cost). Also, the limitation of our proposal opens the ways for advances in the virtualization spectrum on low-cost hardware.

For future work, we will investigate which of applications we execute the infrastructure showcase in this work and measure their performance. This work will bring a better understanding of the supported use cases and will provide information about the limitations associated with such scenarios, thus bringing knowledge about the weaknesses and strengths of the use of low-cost hardware on edge.

ACKNOWLEDGEMENTS

This work is inside UFU-CAPES.Print Program. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. This research also received the support from PROPP/UFU.

REFERENCES

Ahmad, M., Alowibdi, J. S., and Ilyas, M. U. (2017). viot: A first step towards a shared, multi-tenant iot infrastructure architecture. In *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 308–313.

Akrivopoulos, O., Zhu, N., Amaxilatis, D., Tselios, C., Anagnostopoulos, A., and Chatzigiannakis, I. (2018). A Fog Computing-Oriented, Highly Scalable IoT Framework for Monitoring Public Educational Buildings. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–6.

Barik, R. K., Lenka, R. K., Rao, K. R., and Ghose, D. (2016). Performance analysis of virtual machines and containers in cloud computing. In *2016 International Conference on Computing, Communication and Automation (ICCCA)*, pages 1204–1210.

Bekaroo, G. and Santokhee, A. (2016). Power consumption of the raspberry pi: A comparative analysis. In *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies (EmergiTech)*, pages 361–366.

Bellavista, P., Chessa, S., Foschini, L., Gioia, L., and Girolami, M. (2018). Human-Enabled Edge Computing: Exploiting the Crowd as a Dynamic Extension of Mobile Edge Computing. *IEEE Communications Magazine*, 56(1):145–155.

Chang, H., Hari, A., Mukherjee, S., and Lakshman, T. V. (2014). Bringing the cloud to the edge. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 346–351.

d. S. Marques, W., d. Souza, P. S. S., Rossi, F. D., d. C. Rodrigues, G., Calheiros, R. N., d. S. Conterato, M., and Ferreto, T. C. (2018). Evaluating container-based virtualization overhead on the general-purpose iot platform. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 00008–00013.

Gouareb, R., Friderikos, V., and Aghvami, A. (2018). Virtual network functions routing and placement for edge cloud latency minimization. *IEEE Journal on Selected Areas in Communications*, 36(10):2346–2357.

Ha, K. and Satyanarayanan, M. (2015). Openstack++ for cloudlet deployment. *School of Computer Science Carnegie Mellon University Pittsburgh*.

Hu, Y. C., Patel, M., Sabella, D., Sprecher, N., and Young, V. (2015). Mobile edge computing—a key technology towards 5g. *ETSI white paper*, 11(11):1–16.

Kristiani, E., Yang, C.-T., Wang, Y. T., and Huang, C.-Y. (2019). Implementation of an edge computing architecture using openstack and kubernetes. In Kim, K. J. and Baek, N., editors, *Information Science and Applications 2018*, pages 675–685, Singapore. Springer Singapore.

Lei, L., Xiong, X., Hou, L., and Zheng, K. (2018). Collaborative Edge Caching through Service Function Chaining: Architecture and Challenges. *IEEE Wireless Communications*, 25(3):94–102.

Li, Z., Kihl, M., Lu, Q., and Andersson, J. A. (2017). Performance overhead comparison between hypervisor and container based virtualization. In *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, pages 955–962.

Milojičić, D., Llorente, I. M., and Montero, R. S. (2011). Opennebula: A cloud management tool. *IEEE Internet Computing*, 15(2):11–14.

Pahl, C., Helmer, S., Miori, L., Sanin, J., and Lee, B. (2016). A container-based edge cloud paas architecture based on raspberry pi clusters. In *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pages 117–124.

Paolino, M., Rigo, A., Spyridakis, A., Fangude, J., Lalov, P., and Raho, D. (2015). T-kvm: A trusted architecture for kvm arm v7 and v8 virtual machines securing virtual machines by means of kvm, trustzone, tee and selinux. In *In Proc. of the Sixth International Conference on Cloud Computing, GRIDs, and Virtualization*.

Pierre, G. and Stratan, C. (2012). Conpaas: A platform for hosting elastic cloud applications. *IEEE Internet Computing*, 16(5):88–92.

Puthal, D., Obaidat, M. S., Nanda, P., Prasad, M., Mohanty, S. P., and Zomaya, A. Y. (2018). Secure and Sustainable Load Balancing of Edge Data Centers in

Fog Computing. *IEEE Communications Magazine*, 56(5):60–65.

Rodola, G. (2016). Psutil package: a cross-platform library for retrieving information on running processes and system utilization.

Sallent, S., Abelém, A., Machado, I., Bergesio, L., Fdida, S., Rezende, J., Azodolmolky, S., Salvador, M., Ciuffo, L., and Tassiulas, L. (2012). Fibre project: Brazil and europe unite forces and testbeds for the internet of the future. In Korakis, T., Zink, M., and Ott, M., editors, *Testbeds and Research Infrastructure. Development of Networks and Communities*, pages 372–372, Berlin, Heidelberg. Springer Berlin Heidelberg.

Schiller, E., Nikaein, N., Kalogeiton, E., Gasparyan, M., and Braun, T. (2018). CDS-MEC: NFV/SDN-based Application Management for MEC in 5g Systems. *Computer Networks*, 135:96 – 107.

Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646.

Silva, A. P., Tranoris, C., Denazis, S., Sargento, S., Pereira, J., Luís, M., Moreira, R., Silva, F., Vidal, I., Nogales, B., Nejabati, R., and Simeonidou, D. (2019). 5ginfire: An end-to-end open5g vertical network function ecosystem. *Ad Hoc Networks*, 93:101895.

Silva, F. S. D., Lemos, M. O. O., Medeiros, A., Neto, A. V., Pasquini, R., Moura, D., Rothenberg, C., Mamatas, L., Correa, S. L., Cardoso, K. V., Marcondes, C., ABelem, A., Nascimento, M., Galis, A., Contreras, L., Serrat, J., and Papadimitriou, P. (2018). Necos project: Towards lightweight slicing of cloud federated infrastructures. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pages 406–414.

Tong, L., Li, Y., and Gao, W. (2016). A hierarchical edge cloud architecture for mobile computing. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9.

van Kempen, A., Crivat, T., Trubert, B., Roy, D., and Pierre, G. (2017). Mec-conpaas: An experimental single-board based mobile edge cloud. In *2017 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, pages 17–24.

von Leon., D., Miori., L., Sanin., J., Ioini., N. E., Helmer., S., and Pahl., C. (2018). A performance exploration of architectural options for a middleware for decentralised lightweight edge cloud architectures. In *Proceedings of the 3rd International Conference on Internet of Things, Big Data and Security - Volume 1: IoTBDS,*, pages 73–84. INSTICC, SciTePress.

Zhang, G. and Ravishankar, M. (2019). Exploring vendor capabilities in the cloud environment: A case study of alibaba cloud computing. *Information & Management*, 56(3):343 – 355.

Zhang, J., Hu, X., Ning, Z., Ngai, E. C. ., Zhou, L., Wei, J., Cheng, J., and Hu, B. (2018). Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks. *IEEE Internet of Things Journal*, 5(4):2633–2645.

Zhu, C., Pastor, G., Xiao, Y., and Ylajaaski, A. (2018). Vehicular Fog Computing for Video Crowdsourcing: Applications, Feasibility, and Challenges. *IEEE Communications Magazine*, 56(10):58–63.