This paper was accepted for publication in the Journal of Grid Computing. The copyright was transferred to Springer. The final version of the paper will be made available on Springer Link via https://doi.org/10.1007/s10723-021-09545-3

Opportunistic Deployment of Distributed Edge Clouds for Latency-critical Applications

Eduardo Huedo[a,b,*], Rubén S. Montero[a], Rafael Moreno-Vozmediano[a], Constantino Vázquez[c], Vlastimil Holer[c], Ignacio M. Llorente[a,b]

[a] *Computer Science Faculty, Universidad Complutense de Madrid, Spain.*
[b] *Institute of Knowledge Technology, Universidad Complutense de Madrid, Spain.*
[c] *OpenNebula Systems.*

Abstract
The growing number of latency-critical applications are posing novel challenges for network operators, cloud/hosting companies, and application providers. Edge Computing is the strongest candidate for providing low-latency responses, but it is not yet clear what edge infrastructures will be like. This paper introduces a new platform for enabling an edge infrastructure according to a disaggregated distributed cloud architecture and an opportunistic model based on bare-metal providers. Results from a multi-server online gaming application deployed in a real geo-distributed edge infrastructure show the feasibility, performance and cost efficiency of the solution.

Keywords
Edge Computing, Cloud Computing, Latency, Bare-metal provider, Cloud Disaggregation, Online Gaming

1. Introduction

In 2020, there are now over six billion smartphone users globally and over 50 billion smart-connected devices in the world, all developed to collect, process, and share data. Many companies, both long-established and newly emerging, from multiple industries including gaming, the Internet of Things (IoT), social networking and telecommunications, are focusing their business strategies on being able to provide their customers with innovative services and capabilities with absolute immediacy, in order to process the explosive growth in data traffic.

Latency, or data transfer time, is becoming a critical factor for forthcoming technologies. New applications and uses are emerging that require high-performance real-time responses. Applications such as vehicular control, augmented reality, virtual reality, gaming and new services for industries are turning network requirements upside-down. What is considered to be acceptable latency strongly depends on the application, from 150 milliseconds for voice communication, to less than 10 milliseconds for assisted-driving, and even less than 2 milliseconds for robotics.

Edge Computing [1] is aimed at providing low-latency services and reducing network traffic. The idea of Edge Computing has arisen simultaneously in three different areas: telecommunications, the cloud and the IoT. In the telecommunications industry, the term "edge" traditionally refers to base stations or access networks (e.g., Internet Service Providers; ISP). For cloud providers, this even includes other networks a few hops away from the end user or device. However, in IoT it typically refers to on-premises networks where sensors and IoT devices are located. Therefore, the term "edge" means different things to different people, from a data center serving a region or country, to a computer or computing cluster serving a city, block, street, building, office or room.

These three areas have promoted the development and expansion of Edge Computing through many different technologies and initiatives. For example, Multi-access (or Mobile) Edge Computing (MEC) [2] is an effort by the telecommunications industry to provide ultra-low latency and high bandwidth access to computing services located on mobile base stations. MEC is fundamental for the 5G architecture, where operators can open their access networks to authorized third-parties. CORD (Central Office Re-architected as a Datacenter) is another initiative by the Telecom industry that leverages NFV (Network Function Virtualization), SDN (Software Defined Networking), open-source software, and cloud services to redefine central offices and offer edge services [3].

Many cloud vendors provide their services in tens of independent geographical locations referred to as regions. They also provide services on the edge, mainly for CDN (Content Distribution Networks). Some even offer connections to their services from network edge locations across the globe, known as Points of Presence (PoP).

In IoT, Fog Computing [4] arose to support Cloud Computing, closer to things in the physical world, providing computing capabilities on premises or in network equipment like routers or wireless access points. The Industrial Internet Consortium, which brings together the organizations and technologies necessary to accelerate the growth of the industrial IoT and where the OpenFog Consortium merged, now wants Edge Computing to embrace terms like fog, mist, cloudlets and thing-to-cloud continuum, each with its own particular perspectives and technologies.

* Corresponding author.
E-mail address: ehuedo@fdi.ucm.es (E. Huedo).

Edge Computing opens up new avenues, not only for large telecom companies or cloud providers, but also for small and medium-sized companies and public institutions, allowing them to offer their customers new or improved online services, to meet the increasing demand for low-latency applications, including media streaming, online games, virtual and augmented reality, real-time e-healthcare and autonomous driving, among many others.

However, most of these companies and institutions cannot afford to provide their own physical edge infrastructure, with several geographically distributed PoPs, to satisfy the demands of different users located in various locations. As an alternative, we propose the use of an opportunistic approach, enabling these companies to deploy on-demand private edge cloud infrastructures by dynamically leasing bare-metal resources from third-party providers, such as telecom companies, network operators, or specialized edge providers. This approach offers an opportunistic, agile, flexible, scalable and pay-per-use approach to building Edge Computing environments. Moreover, the use of bare-metal servers involves several benefits compared to virtual servers such as OS independence, full customization capabilities, full hardware control, complete isolation, and fine performance tuning. And, even more importantly, it is a practical and affordable way to put resources on the edge without imposing any special requirements or configuration, and these resources remain open for any current and future use with minimum maintenance.

To efficiently build and manage this opportunistic edge environment, we propose a distributed edge cloud architecture implementing a disaggregated approach [5] that includes edge nodes as components of a geo-distributed private cloud. New edge resources are leased on-demand from existing bare-metal cloud providers and automatically installed and configured with the software stack needed to enroll them in the private edge cloud instance. Then a cloud management platform manages both the physical resources of the on-premises cloud data center, if any, along with the resources of multiple, highly dispersed edge nodes, providing users with uniform access to this disaggregated resource pool. The cloud platform deploys, manages, orchestrates and auto-scales virtual Edge Computing environments onto this geo-distributed physical infrastructure, providing different virtualization technologies such as virtual machines (VMs) or system containers, according to the service needs.

This paper introduces a new platform for enabling an Edge Computing environment according to an opportunistic model based on bare-metal providers. To this end, the OpenNebula cloud manager has been extended with a new component, OpenNebula Provision, which allows users to manage the full life cycle of the physical infrastructure at different edge locations, enabling the deployment of nodes on the edge. The paper also analyzes security, monitoring, networking and image transfer challenges, and demonstrates the feasibility and potential of the platform when executing a multi-server online gaming application over a real geo-distributed edge infrastructure built on-demand with physical resources from a commercial cloud provider. The structure of the article is as follows: Section 2 describes latency-critical applications; Section 3 reviews the related work; Section 4 provides an architectural description of the platform and its components, including the new OpenNebula Provison component; Section 5 presents the experimental results collected from a real geo-distributed edge infrastructure with a real gaming application; and, finally, Section 6 provides some conclusions.

2. Latency-critical Applications

A growing number of applications demand low (below 20 ms) and ultra-low (below 5 ms) latencies, posing novel challenges for network operators, cloud/hosting companies, and application providers. These applications, as shown in Fig. 1, may be from various fields, including consumer applications, industrial applications, e-health services, and smart mobility applications.

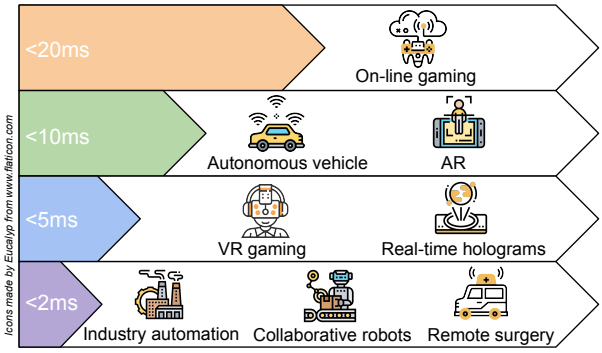

Fig. 1. Latency-critical applications.

Within the consumer applications market, online gaming is a good example of consumer application with low latency requirements (< 20 ms). The gaming population is undoubtedly expanding and evolving, investing more of their time in playing, as well as demanding increasingly more from their gaming experience. Not only do gamers look for tantalizing graphics, but they expect a comprehensive, life-like experience, which oftentimes translates into "speed-of-thought" response times. Consequently, while online gaming companies are developing more advanced games with increasingly

data-intensive processing, they are presented with the conflicting reality of being able to distribute this huge amount of data to their players with minimum latency. The current entertainment industry is also focused on enhancing user experience by deploying interactive applications with virtual or augmented reality (VR, AR) elements, which are also very latency-sensitive (< 10 ms). AR services, for instance, need an application to be able to analyze, in real time, the output from a device's camera, combine that information with data associated with a specific location, and then generate the view that the end user will obtain when visiting that specific point of interest. Some ongoing consumer applications also have ultra-low latency requirements (< 5 ms), such as real-time holography transmission or haptic/tactile VR games, where the system's feedback must be adapted to human reaction times.

Within the automotive industry, smart mobility applications, including autonomous driving, are becoming a reality. The response of an autonomous vehicle to an unexpected obstacle (e.g., a pedestrian crossing the street) needs to be immediate. In practical terms, it is neither feasible nor scalable to locate all the necessary decision-making processing power within the remote infrastructure of a centralized cloud. Given the amount of data that autonomous cars generate, and their dependence on quick analyses and low response times (< 10 ms), dealing with that volume of data will certainly require Edge Computing. In this case, safety requirements have to be taken into account to avoid or reduce physical damage.

With the digitalization of the manufacturing sector came an increase in the use of automation, software and data analytics, as well as vast amounts of data. IT (Information Technology) and OT (Operational Technology) convergence has now become a crucial part of this new scenario. Edge Computing facilitates platforms for this convergence, enabling manufacturers to move away from closed, proprietary solutions. Manufacturers will be able to take full advantage of industrial IoT devices and put in place new applications that will be in charge of modifying production processes on the basis of real-time information, thus adapting more quickly to the current environment and reacting almost instantly to unexpected situations. Factory automation can involve different time-critical processes with ultra-low latency requirements (< 2 ms), such as real-time closed-loop robotic control, video-driven machine-human interaction, and AR applications for training and maintenance.

There are also a good number of e-health applications that require low and ultra-low latency communications. Patients with chronic conditions are becoming increasingly dependent on wearable healthcare devices that provide real-time monitoring and early warnings, as well as alerting caregivers immediately when help is required. Robotics is also playing a growing role in remote surgery, with devices that must be able to analyze data quickly in order to assist safely and accurately with minimal latency tolerance (< 2 ms). The end results could be fatal if these critical devices relied on decisions where data was sent to a centralized cloud, especially where large volumes of information are involved. Also, dependability mechanisms must be in place to always provide a fast response from the edge.

Many automotive, industrial, and healthcare applications rely on common technologies such as the Internet of Things (IoT), data analytics, and artificial intelligence (AI). The inception of IoT technology has amplified the cloud computing paradigm by establishing the means for distributing computing and processing activities away from the centralized cloud. With the broad explosion of smart devices and their compute processing capabilities, technologies and their supporting organizations are focused on utilizing these distributed devices as more than just receptors of processed information, making them active processors of information as well. This is critical to reducing latency and creating a much more "real-life"-like response time, the key to so many emerging technologies. Smart devices that are distributed across the globe cannot effectively depend on a centralized cloud to perform the cumulative analytical workload at the speed needed. The ability of Edge Computing devices and systems to analyze data streaming in almost real time significantly improves awareness of and early response to events that take place across a network. Now, with increasingly sophisticated processing capabilities moving to the edge, these distributed systems can more quickly analyze data for inference and pattern-detention. AI technologies —and Machine Learning in particular— can help to identify patterns and anomalies and can also make predictions based on large datasets. This is proving to be particularly useful for implementing more effective cybersecurity measures.

Edge Computing is being widely adopted as enabling technology to support different types of low-latency applications. Edge Computing can be supported and implemented by a variety of computing and network infrastructures, from customer devices to telecom infrastructures, as shown in Fig. 2. Customer devices (smartphones, smartwatches, laptops, wearable devices, etc.), along with computing capabilities that can be offered by local network devices, such as routers, access points, and so on, constitute the so-called fog computing layer, which can support ultra-low latency applications (< 2 ms). SMEs, office buildings, research and educational institutes, among others, can also host their own on-premise infrastructures to offer local services and computing resources to local customers. Some kinds of ultra-low latency services (< 5 ms) can be deployed and implemented in these physical premises. However, since the computing capabilities of customer devices and customer premises can be limited, these local resources may be complemented by external resources leased from third party providers, such as telecom companies, ISPs, or edge infrastructure providers, which implement micro-data centers at the edge of the network, offering on-demand resources in the form of bare-metal hosts or virtual servers, and providing very limited latencies (5-20 ms). In addition, centralized cloud providers can offer almost unlimited out-of-the-edge computing capabilities to support non-critical latency services.

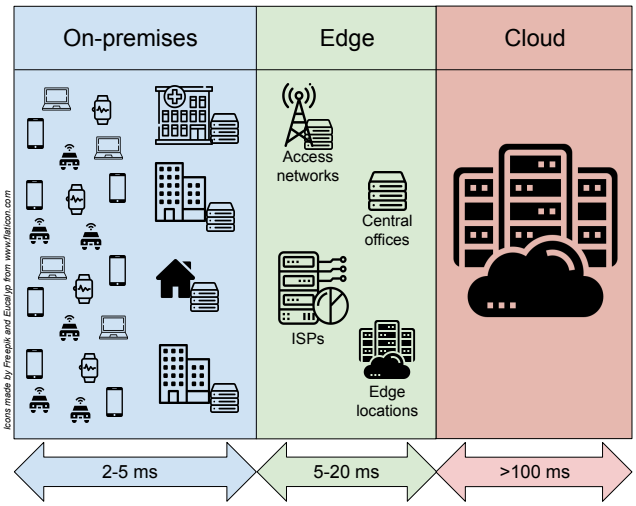

Fig. 2. Supporting infrastructures for Edge Computing.

The proposed edge cloud platform is aimed at managing both resources from customer premises (if any), and bare-metal edge resources from third party providers (customer devices are beyond the scope of this work), allowing the deployment of a disaggregated and opportunistic private Edge Computing environment that supports low latency applications with several geographically distributed PoPs.

3. Related Work

Edge Computing has received a great deal of research interest since Satyanarayanan *et al*. [6] proposed an architecture based on cloudlets, which are small-scale data centers placed on the edge to support resource-intensive and interactive mobile applications or the IoT.

Multi-access Edge Computing (MEC) using 5G mobile networks is a hot topic in both research and industry, with prototypes being deployed at an increasing scale. For example, Chirivella *et al*. [7] set out a novel 5G service deployment orchestration architecture capable of automating and coordinating a series of complicated operations across physical infrastructure, virtual infrastructure, and service layers over a distributed MEC infrastructure. This architecture was validated through experiments on a controlled testbed where 5G services were installed from bare-metal servers, obtaining service deployment times of between 36 and 56 minutes.

Most Edge Computing platforms are based on a distributed model, where each edge node is independently managed. These distributed models include an upper-level edge gateway responsible for interacting with clients that routes client requests to the appropriate bottom edge nodes. For example, OpenStack StarlingX [8] provides a container-based platform to build mission critical edge clouds, and the Distributed Cloud subproject supports an edge computing solution by providing central management and orchestration for a geographically distributed network of StarlingX Kubernetes edge clusters, which are centrally managed and synchronized over L3 networks from a central cloud. Also, KubeEdge [9] enables the orchestration and management of edge clusters similar to how Kubernetes manages in the cloud. Other interesting research focusing on the management of distributed fog computing infrastructures is ENORM [10], an edge node resource management framework based on a three-tier fog architecture (device tier, edge node tier, and cloud tier). Each edge node in ENORM is provided with various components that implement different management capabilities. These distributed solutions are highly scalable; however, they delegate most complex management tasks to the edge nodes, which can consume a lot of their resources. Furthermore, installing, configuring, and maintaining the edge nodes can be an extremely complex and time-consuming task.

In contrast, the disaggregated model is based on a centralized management, which releases edge nodes from complex management tasks, showing a good tradeoff of performance and design complexity, and provides a uniform view of all the distributed edge infrastructure [5]. The main disadvantage of this model is its scalability, which is limited with regard to the number of edge nodes that can be monitored and managed.

There is also extensive research into latency-sensitive applications. For example, Maheshwari *et al*. [11] analyzed the scalability and performance of a city-scale edge cloud system designed to support latency-sensitive applications. Ascigil *et al*. [12] proposed a set of practical, uncoordinated strategies for service placement in edge clouds deployed by ISPs and evaluated these through simulations. Yi *et al*. [13] presented an Edge Computing platform to provide low-latency video analytics closer to the users.

For online games, Dick *et al*. [14] analyzed various factors that affect a player's perception and performance in multiplayer games. They reported that a one-way delay of 80 milliseconds, on average, was acceptable for most users of First Person Shooter (FPS) games, while Role Playing Games (RPG) and Real Time Strategy (RTS) game users can accept one-way delays of up to 120 and 200 milliseconds, respectively [15].

The use of Edge Computing to reduce latency in games has also been studied. For example, Choy *et al*. [16] presented simulation studies to determine the percentage of end-users served by game servers. Using the cloud, only 70% of end-users achieved the latency target (80 milliseconds). With an edge-only deployment, 90% of end-users could be served, but an excessive number of edge servers would be required. For this reason, they proposed an alternative architecture that complemented cloud servers with edge servers deployed using CDNs co-located at ISP sites.

Cloud gaming consists of rendering games on cloud servers and streaming them to thin clients [17, 18]. Tian *et al*. [19] presented a simulation study for cost-effective cloud gaming using geo-distributed data centers. Lin *et al*. [20] suggested a lightweight fog-based system for cloud gaming where the system comprises idle machines that are close to the end-users and which connect to the cloud. The intensive computation involved in the new game state of the virtual world is conducted in the cloud, which then sends update messages to fog nodes, while fog nodes update the virtual world, render game videos, and stream videos to the players. In this case, real experiments on PlanetLab were conducted in addition to simulations.

The proposed edge platform differs from previous approaches in that it follows a disaggregated model, provides a feasible, agile and cost efficient solution, and is available to experiment with real applications in real geo-distributed infrastructures.

4. Architecture and Selected Components

In this section, we briefly discuss the main components of an edge cloud infrastructure and their functional aspects, based on the ONEedge project (www.oneedge.io) [21], where OpenNebula Systems (the company behind the OpenNebula cloud manager) is involved. This project aims to provide an automated software-defined platform to build private Edge Computing platforms based on resources leased on demand in close proximity to the users and devices. We use a layered approach to architecturally describe the system, which is depicted in Fig. 3.

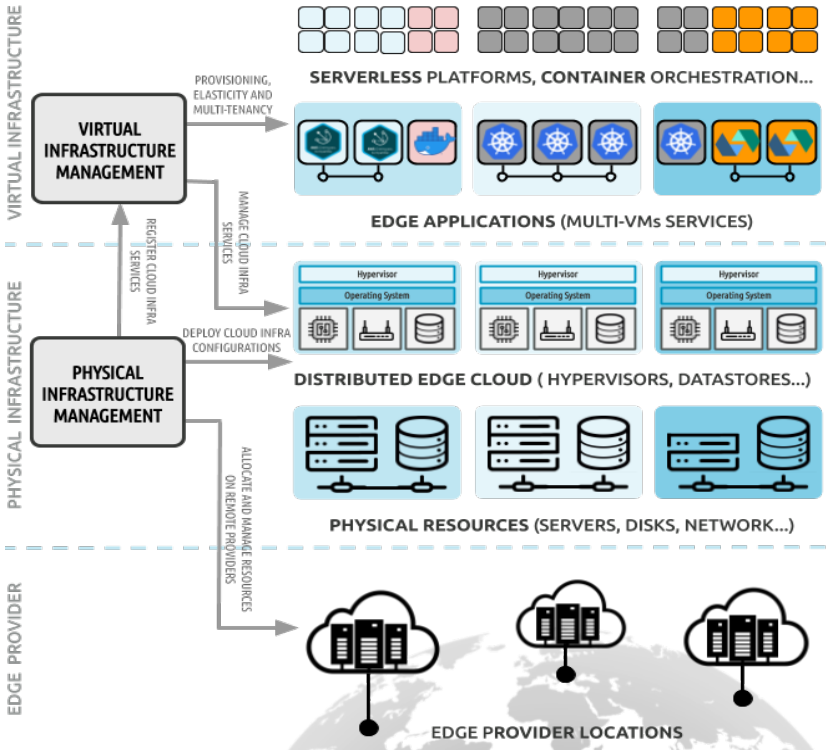

Fig. 3. Main components and layers of a distributed edge architecture.

Each layer is characterized by a well-defined API and one or more management components, namely (from the bottom up):

1. *Edge Provider*. This layer consists of provider companies with data-centers distributed across multiple edge locations. The ecosystem of providers is heterogeneous in terms of pricing schemes, API, capacity offer, provision process and features. However, we can abstract this heterogeneity assuming minimal functional requirements including: bare-metal server, IP network and storage provision. We envision the proliferation of marketplaces that aggregate the offer of this collection of providers.
2. *Physical Infrastructure*. This layer consists of the hosts, storage and networks provisioned at edge locations plus the required runtime components (e.g., hypervisors) and configurations (e.g., storage volume mount) needed to run the application workload. In this layer, the physical infrastructure manager is responsible for interfacing with

the edge providers to deploy the resources and perform the required customizations. Additionally, this component manages the state and life cycle of the previous components.

3. *Virtual Infrastructure*. This layer comprises the abstractions needed to deploy applications on a virtualized infrastructure, including: virtual networks, virtual disk images and VMs. In this layer, the virtual infrastructure manager is responsible for managing these virtual resources including their life cycle, as well as implementing capacity allocation policies and enforcing access policies in a multi-tenant environment.
4. *Applications*. This layer consists of edge applications and services, including support tools like application marketplaces, serverless platforms and container orchestration tools.

The next three subsections describe the selected components for this architecture, namely Packet (www.packet.com) as the edge provider, OpenNebula Provision as the physical infrastructure manager and OpenNebula Cloud Manager as the virtual infrastructure manager. A fourth subsection describes the application layer that, for experimental purposes, is based on an online multiplayer FPS game. The last subsection highlights the main challenges in edge cloud infrastructure management compared to standard cloud deployment.

4.1. Edge Provider

In general, an Edge Provider is an entity that provides content, services, applications or raw computing resources accessed over the Internet and located at the *edge* of the ISP backbone network [22]. These providers may range, in a broad sense, from video streaming or gaming services, to media platforms or bare metal servers. There is a tight relationship between the edge and the ISP and mobile carriers that deliver the edge content to the users. This relationship has caused interesting dynamics in the edge market: ISPs are becoming edge providers and big edge providers are deploying their own access networks to become ISPs. This complex ecosystem is beyond the scope of this work, where we will concentrate on bare-metal edge providers.

We chose the Packet provider platform to deploy our edge cloud infrastructure. Packet is a bare-metal resource provider focused on bringing the cloud experience to physical infrastructure, regardless of what it is and where it resides. This provider manages tens of thousands of physical servers, built by a dozen different manufacturers, across three architectures and over 20 global facilities. It offers 7 core locations and a growing number of edge locations, which are small data centers in metropolitan areas, some of them deployed at or near wireless towers. Many official operating systems are supported and custom OS images can be installed using iPXE, an open-source implementation of the Preboot eXecution Environment (PXE) client firmware and bootloader. With its focus on bare-metal servers, fast provisioning, and distributed locations, Packet is a consummate platform for building out the edge.

4.2. Physical Infrastructure

In the context of the ONEedge project [21], we have developed OpenNebula Provision, which is a new component that allows users to manage the full life cycle of the physical infrastructure at completely independent edge locations, starting with their provision and maintenance. An edge provision is a declarative definition (provision template) of the infrastructure resources required at each location. A location is defined as a group of physical hosts allocated from the remote bare-metal edge provider. These are fully configured with the edge hypervisor technology and enabled in the OpenNebula stack. Additionally, each provision may include dedicated virtual networks and datastores. OpenNebula Provision follows a modular design that uses provider drivers to allocate resources in different edge provider locations.

Each provision run prepares a new ready-to-use dedicated OpenNebula cluster, with its datastores, virtual networks and hosts, in one location. Hosts and VMs are started from the base OS images and configured dynamically from scratch. There is no need to pre-install images or pre-configure services. The use of custom OS images with iPXE will be considered in future work. Each run performs the following phases, fully automatically:

1. Provision: A new physical server is provisioned from Packet, with a clean installation of the chosen OS. The server is configured as a host in an OpenNebula cluster, but it is disabled.
2. Configuration: The KVM hypervisor is installed and configured on the physical server.
3. Enablement: The OpenNebula host is enabled, meaning that once the needed drivers are transferred, OpenNebula starts monitoring and managing the host.

4.3. Virtual Infrastructure

OpenNebula is an open source management platform for building private, public and hybrid IaaS clouds. It orchestrates computing, storage and network resources to provide a uniform management layer for VMs and infrastructure containers in a multi-tenant environment.

OpenNebula incorporates the components and enhancements needed to meet the requirements of edge infrastructures. OpenNebula Cloud Manager is responsible for orchestrating the virtual edge infrastructure resources described above, and managing the life cycle of the application instances running on the edge infrastructure. In addition, its VM contextualization mechanism allows the automation of various VM configuration aspects at boot time, such as network

configuration, passing user credentials or security tokens, or executing startup scripts to install some packages or start a given service. OpenNebula Cloud Manager is integrated with OpenNebula Provision, which registers or unregisters the cloud infrastructure services deployed across the edge locations.

Fig. 4 shows the main components of OpenNebula Cloud Manager. In this image we highlight those components that are impacted by the specific characteristics of an edge infrastructure, compared to a *classical* cloud infrastructure, as we will discuss in Subsection 4.5.

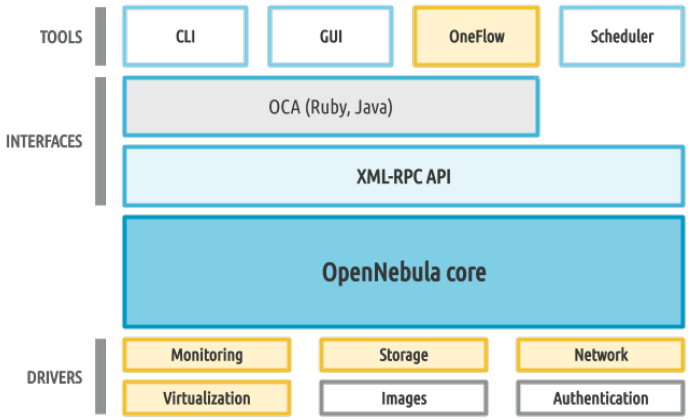

Fig. 4. OpenNebula architecture and main components.

4.4. Applications

As described in Section 2, there are many latency-sensitive applications that can benefit from Edge Computing: from 5G services to IoT gateways or responsive services. Fig. 5 depicts the structure of edge applications. The simplest edge application or service consists of a single component, implemented by one VM or system container. In general, edge applications and services consist of more than one component with deployment order constraints and elasticity rules. Each application component is implemented by a VM or system container, which is defined by: the application image, its network requirements, a capacity specification and optional hardware constraints. Elasticity rules and deployment order constraints for multi-component applications may also be specified.

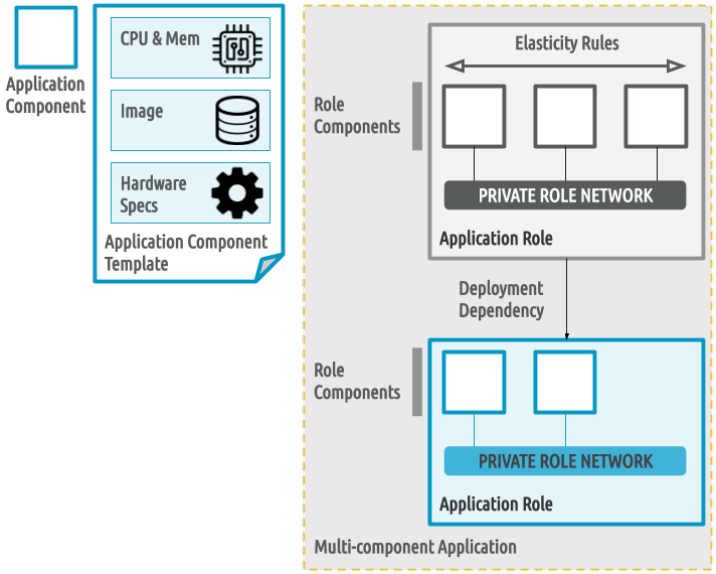

Fig. 5. Edge application deployment.

Since this work is focused on performance, we chose a multiplayer online game for the experiments, because it is the type of application where latency reduction is critical for the user experience, but it does not have a big impact on human life (like autonomous driven has, for example), since it cannot cause physical damage. The gaming application chosen is "Enemy Territory: Legacy", an open source project that provides a compatible client (and server) for the game "Wolfenstein: Enemy Territory". It is an online multiplayer FPS game based on the Quake 3 game engine, but which incorporates more complex gaming rules. We chose this service for its maturity and simplicity and also because, as mentioned earlier, FPS games are more sensitive to network latency than other game genres, such as RPG or RTS.

Enemy Territory, like many other online games, is based on a client-server architecture with several clients (users) connected to a server that handles the game logic. Each client collects the actions from the user, such as movements, shots, and so on, and sends these to the server. The server validates and processes the client inputs, computes the next game state, and sends this back to the client. The client then renders the received game state on the local display, and the process repeats. The main sources of delay in this game logic are the upload delay when sending updates from client to server, the server processing delay, the download delay when sending the new state from server to client, and the client processing delay when rendering. While processing delays depend on the client and server computing capacity, upload and download delays depend on the network connection characteristics between client and server. To reduce network latency, the game service can be deployed on several servers in different locations, so that the client can choose the server with the lowest latency based on ping probes. Network traffic is based on UDP packets. The typical network traffic profile for this game is shown in Table 1 [23].

	Server to client	**Client to server**
Packet length	40-300 bytes	55-72 bytes
Inter arrival time	45-55 ms	10-70 ms
Packet rate	20 packets/s	20-90 packets/s
Data rate	12-30 Kbps	10-45 Kbps

Table 1. Typical network traffic profile for Enemy Territory.

4.5. Challenges and Improvements

The main difference in edge infrastructure management arises from the fact that the hosts are distributed across different networks and accessed through public Internet and unreliable connections. This model deviates from that of the traditional private cloud based on on-premises resources interconnected through private, high-performing networks. The edge architecture requires secure communication in the control plane, where every operation on the edge nodes needs to be initiated through a secure, authenticated channel. OpenNebula monitor drivers have been extended to send encrypted monitor and status information to the OpenNebula front-end. Also, some reliability mechanisms have been put in place to control connection errors, in both the transport (i.e. use of TCP transport) and application layers (i.e. transmission of cumulative data), for the different message types.

Virtual network management needs to implement abstraction mechanisms to interface each edge provider. Specific IP Address Management (IPAM) components are usually required to register public IP ranges and route these to the edge host in the provider network. The host also needs to route the public traffic to the application component using Network Address Translation (NAT) techniques. Finally, some network-specific use cases require the deployment of Virtual Network Functions (VNF) on the edge. OpenNebula network drivers have been integrated with the network stack of Packet and AWS providers to dynamically allocate and configure associated private and elastic IPs when the virtual application needs them.

Another significant aspect is the distribution of application images and runtime components across the edge locations. Considering that application images are typically files ranging from hundreds of MB to tens of GB, which follow standard upgrade and patching procedures, a caching mechanism to distribute and synchronize images to the edge nodes has been added to OpenNebula to efficiently deploy applications hiding the network bandwidth. Storage drivers in OpenNebula have also been improved to preserve sparse file gaps to not increase file sizes and hence transfer times.

Multi-component applications may need to incorporate location-aware components (e.g., GeoDNS or anycast) to perform global load balancing across locations, or even give the application the ability to guide the location selection process, with location-specific elasticity rules for both application components (managed by OpenNebula) and physical infrastructure components (managed by OpenNebula Provision).

5. Experiments

To evaluate the feasibility, functionality, performance and cost-efficiency of the proposed opportunistic model for distributed edge clouds, we performed a set of experiments in a real environment by deploying a typical latency-sensitive application, a multi-server online gaming application, over a disaggregated infrastructure.

The infrastructure used for these experiments, from the Packet bare-metal provider, is described in Table 2, and its geographic distribution, involving 5 core and 12 edge locations all over the world, is depicted in Fig. 6. OpenNebula Provision and OpenNebula Cloud Manager, not including the improvements described in Section 4.5, were installed on a server in Parsippany (EWR1). Two hardware configurations (x1.small.x86 and c2.medium.x86) were used with the Ubuntu 18.04 LTS operating system and KVM hypervisor.

As described in Section 4, each provision execution prepares one cluster in a single location in three different phases: provision, configuration and enablement. Therefore, several independent provisions against different locations start in parallel. When an OpenNebula cluster is ready, a VM starts, so OpenNebula transfers the base VM image (320 MiB) and

then boots the VM. During the boot process, the game server is installed in the VM, using the OpenNebula VM contextualization mechanism. At the end of the process, there is a fully working game server automatically included in the list of all the other public game servers worldwide.

Name	Location	Type	Hardware
AMS1	Amsterdam, Netherlands	Core	c2.medium.x86
BOS2	Boston, MA USA	Edge	c2.medium.x86
DFW1	Dallas, TX USA	Edge	x1.small.x86
DFW2	Dallas, TX USA	Core	c2.medium.x86
EWR1	Parsippany, NJ USA	Core	c2.medium.x86
FRA2	Frankfurt, Germany	Edge	x1.small.x86
HKG1	Hong Kong, China	Edge	x1.small.x86
IAD1	Ashburn, VA USA	Edge	x1.small.x86
LAX1	Los Angeles, CA USA	Edge	x1.small.x86
MRS1	Marseille, France	Edge	x1.small.x86
NRT1	Tokyo, Japan	Core	x1.small.x86
ORD2	Chicago, IL USA	Edge	c2.medium.x86
ORD3	Niles, IL USA	Edge	c2.medium.x86
SIN1	Singapore, Singapore	Edge	x1.small.x86
SJC1	Sunnyvale, CA USA	Core	x1.small.x86
SYD1	Sydney, Australia	Edge	x1.small.x86
YYZ1	Toronto, Canada	Edge	x1.small.x86

Table 2. Infrastructure from Packet used in the experiment.

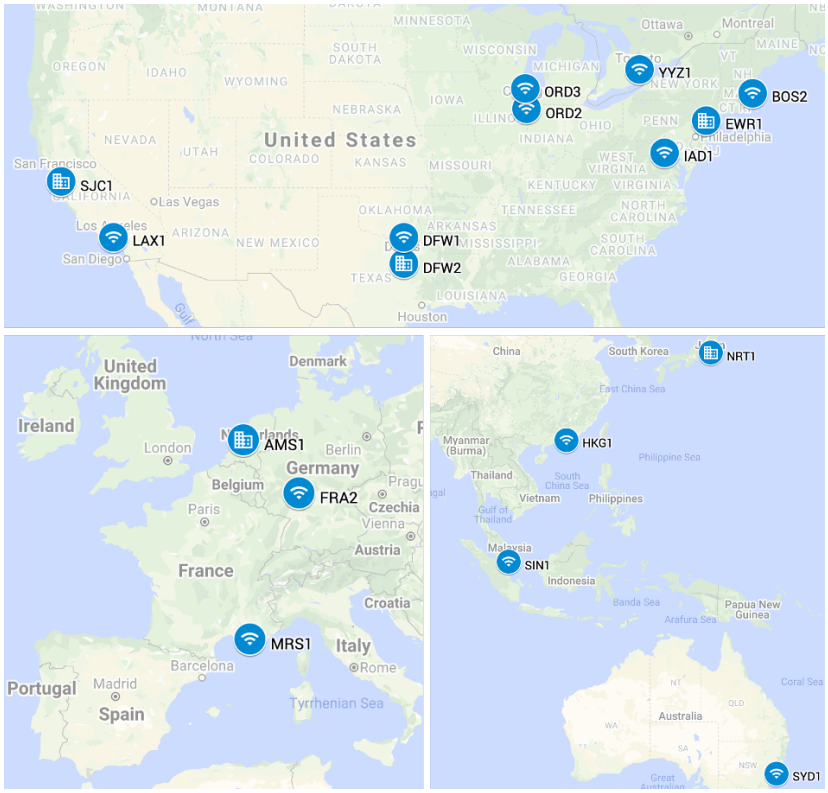

Fig. 6. Geographic distribution of the infrastructure.

The first experiment (in April, 2019) was aimed to demonstrate the feasibility of the approach on a large scale. Therefore, the game server was deployed on all locations. As shown in Table 3 and Fig. 7, the quickest deployment was in Toronto (YYZ1), which took over 8 minutes in total. The slowest deployment was in Hong Kong (HKG1), which took

25 minutes, with most of the time spent on server provision and configuration, as well as VM boot (due to image transfer). To clear out the entire cloud infrastructure, the simultaneous execution of the host deletions only took 49 seconds.

The times for provisioning a server from Packet were quite consistent, with low variance. The time for configuring the KVM hypervisor is affected by network latency between the OpenNebula front-end and the physical server, as well as the performance of the nearest OS package mirrors. The time for transferring the OpenNebula drivers is only affected by network latency and throughput. The same applies for the VM start, as the base image (320 MiB) must be transferred from the OpenNebula front-end to each host. For the distant hosts to the front-end, the image transfer times can be quite long, for example, 2.8 minutes in the case of Sydney (SYD1). During the game service bootstrap, the time depends mainly on the performance of the nearest OS package mirrors.

Name	Provision	Configuration	Enablement	VM start	Service bootstrap	Total time
AMS1	462	163	98	23	82	**831**
BOS2	444	87	11	23	105	**673**
DFW1	309	129	44	15	78	**579**
DFW2	427	125	44	22	86	**707**
EWR1	548	77	3	8	109	**748**
FRA2	346	168	97	21	84	**719**
HKG1	351	352	275	376	146	**1504**
IAD1	305	93	12	32	80	**525**
LAX1	342	172	77	17	96	**707**
MRS1	329	194	104	24	75	**729**
NRT1	350	328	241	46	138	**1105**
ORD2	452	103	24	7	89	**678**
ORD3	438	103	25	36	85	**691**
SIN1	329	424	320	58	108	**1242**
SJC1	341	177	89	19	117	**746**
SYD1	345	462	342	60	171	**1383**
YYZ1	308	102	16	9	82	**520**

Table 3. Times (seconds) for each phase of the deployments.

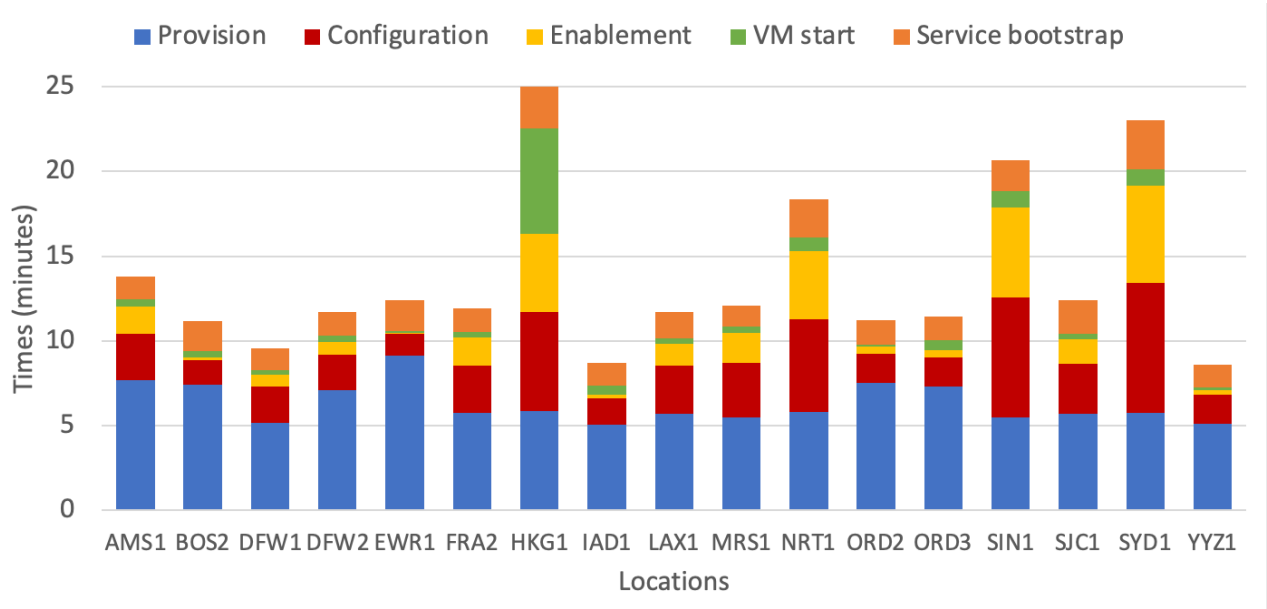

Fig. 7. Times (minutes) for each phase of the deployments.

Fig. 8 presents a screenshot of a client in Brno (Czech Republic) showing the provisioned servers hosting the Enemy Territory video game. The "PING" metric in Fig. 8 provides the network Round Trip Time (RTT) measured from this client to the various game server instances, and Fig. 9 shows the one-way latency derived from this metric.

Understandably, the location with the shortest latency was Frankfurt (FRA2) with 9 milliseconds. In contrast, the latency to Sydney (SYD1) was 165.5 milliseconds (almost 20 times longer).

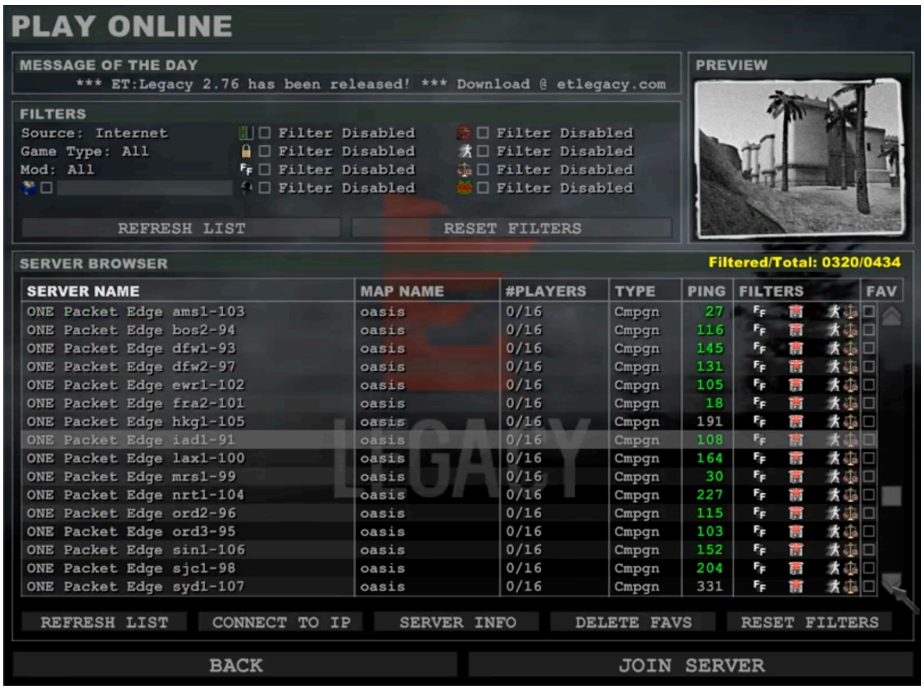

Fig. 8. Game client with the server browser showing the deployed game servers.

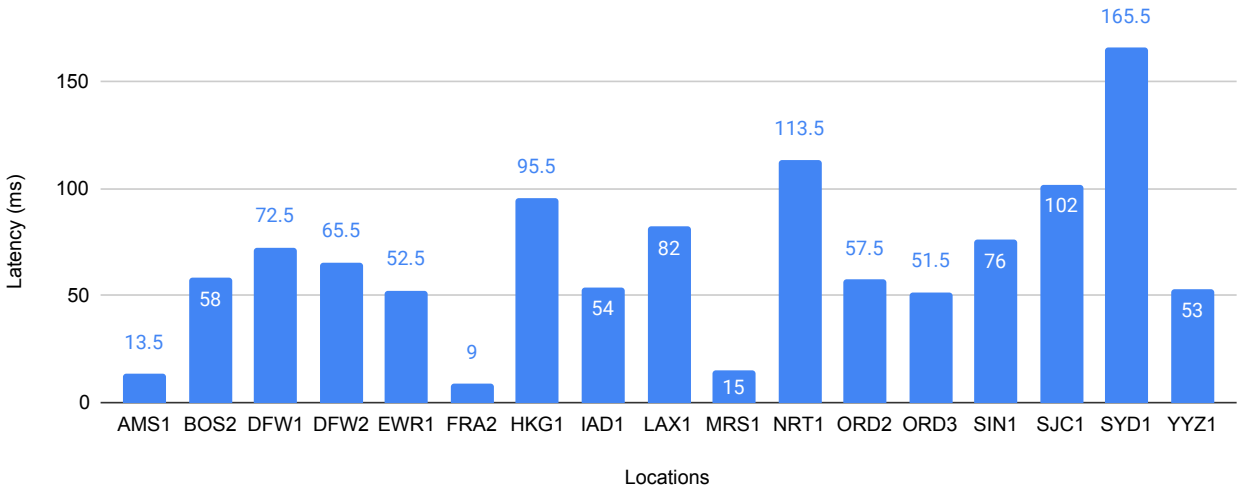

Fig. 9. Latency (milliseconds) to each game server.

With this disaggregated cloud infrastructure using core and edge bare-metal resources, players are directed to the node with the shortest latency, providing optimal performance and user experience. Moreover, the infrastructure operator has the flexibility to scale resources in each cluster according to the active traffic in a cost-efficient way, since in this case the hourly rate for the resources hosting the disaggregated cloud was only $11.40, less than $0.70 per location.

The second experiment (in September, 2020) was aimed to analyze the performance of the approach in more detail. Therefore, the game server was deployed in a subset of locations (those available on the date of the experiment), and 5 runs of the deployment were performed for statistical analysis. There were 9 globally distributed locations (6 in America, 2 in Asia and 1 in Europe) and four of them were of the edge type. In this case, the features for edge infrastructures newly developed in OpenNebula, described in Section 4.5, were used.

As shown in Fig. 10, the variability of the time required for each phase of the deployment is very low in most sites, and so is for the total time. Provision times are generally lower compared to the first experiment (10% overall), due to Packet's optimized procedures for server provisioning and installation. However, the provision takes longer in three locations (IAD1, LAX1 and YYZ1) and, in two of them (IAD1 and LAX1), presents more variability. This could be caused by the demand on that locations in that moment or its operational status, and it also depends on the combination of hardware and OS selected. The time to enable the hosts and start the VMs have been considerably reduced (94% and

56%, respectively), thanks to the improvement of image transfer in OpenNebula. Regarding the times depending on external services, configuration times are lower (17%) while service bootstrap times are higher (19%). The fastest deployment was in EWR1, with 8.2 minutes (53% faster than before). The slowest deployment was in NRT1, with 17.4 minutes (despite being 6% faster than before). The time in HKG1, which was the slowest deployment in the first experiment, has been reduced to 15.8 minutes (59% faster). However, the time in YYZ1, which was the fatest deployment in the first experiment, increased to 10.6 minutes (22% slower) due to higher provision times.

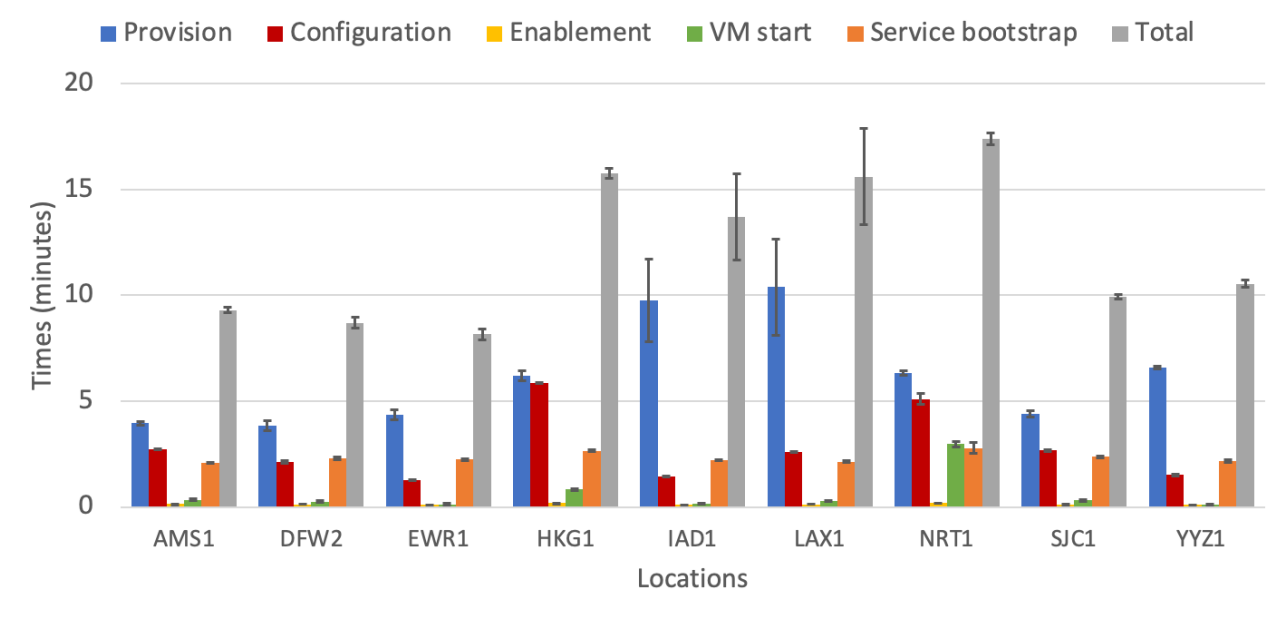

Fig. 10. Time averages and standard deviations (minutes).

6. Conclusions

This article proposes a distributed edge cloud platform implementing a disaggregated approach. Edge resources are leased from existing bare-metal cloud providers and automatically installed, configured and enrolled in a cloud management platform, providing users with a uniform access to this disaggregated resource pool.

The results show the viability of the proposal when deploying an online gaming application across 17 geographically discrete locations in 25 minutes (with the first location ready in less than 9 minutes) for less than $12/hour, achieving the targeted latency of below 10 ms. Additionally, the entire infrastructure can be shut down in less than one minute. These numbers prove the feasibility, performance and cost efficiency of the approach.

Future work involves improving the components of the platform that are impacted by the specific characteristics of edge infrastructures. This includes adding security and reliability, as the infrastructure is spread over public and unreliable networks; improving monitoring to save traffic between the edge and the cloud; implementing a caching mechanism to distribute images to the edge nodes and synchronizing these; designing location-specific elasticity rules for both physical and virtual resources; and exploring different global load-balancing techniques.

Acknowledgments

This work was supported by the Ministerio de Ciencia, Innovación y Universidades through the EdgeCloud research project (RTI2018-096465-B-I00), by the Comunidad de Madrid through the EdgeData research program (P2018/TCS4499), and by the European Union through the ONEedge grant (880412).

References

1. W. Shi, S. Dustdar: The Promise of Edge Computing. Computer 49(5), 78-81 (2016)
2. Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, V. Young: Mobile Edge Computing: A key technology towards 5G. ETSI White Paper No. 11 (2015)
3. R. S. Montero, E. Rojas, A. A. Carrillo, I. M. Llorente: Extending the Cloud to the Network Edge. Computer 50, 91-95 (2017)
4. F. Bonomi, R. Milito, J. Zhu, S. Addepalli: Fog computing and its role in the internet of things. In: Proceedings of 1st workshop on Mobile Cloud Computing (MCC), pp. 13-16 (2012)
5. R. Moreno-Vozmediano, E. Huedo, R. S. Montero, I. M. Llorente: A Disaggregated Cloud Architecture for Edge Computing. IEEE Internet Computing 23(3), 31-36 (2019)

6. M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies: The Case for VM-Based Cloudlets in Mobile Computing. IEEE Pervasive Computing 8(4), 14-23 (2009)
7. E. Chirivella-Perez, J. M. Alcaraz Calero, Q. Wang, J. Gutiérrez-Aguado: Orchestration Architecture for Automatic Deployment of 5G Services from Bare Metal in Mobile Edge Computing Infrastructure. Wireless Communications and Mobile Computing 2018, Article ID 5786936, 1-18 (2018)
8. StarlingX project (https://www.starlingx.io/)
9. Y. Xiong, Y. Sun, L. Xing and Y. Huang: Extend Cloud to Edge with KubeEdge. In: Proceedings of IEEE/ACM Symp. Edge Computing (SEC), 373-377 (2018)
10. N. Wang, B. Varghese, M. Matthaiou and D. S. Nikolopoulos: ENORM: A Framework for Edge NOde Resource Management, IEEE Trans. Services Computing, In press
11. S. Maheshwari, D. Raychaudhuri, I. Seskar, F. Bronzino: Scalability and Performance Evaluation of Edge Cloud Systems for Latency Constrained Applications. In: Proceedings of IEEE/ACM Symp. Edge Computing (SEC), pp. 286-299 (2018)
12. O. Ascigil, T. K. Phan, A. G. Tasiopoulos, V. Sourlas, I. Psaras, G. Pavlou: On Uncoordinated Service Placement in Edge-Clouds. In: 2017 IEEE Intl. Conf. Cloud Computing Technology and Science (CloudCom), pp. 41-48 (2017)
13. S. Yi, Z. Hao, Q. Zhang, Q. Zhang, W. Shi, Q. Li: LAVEA: latency-aware video analytics on edge computing platform. In: Proceedings ACM/IEEE Symp. Edge Computing (SEC), Article 15, pp. 1-13 (2017)
14. M. Dick, O. Wellnitz, L. Wolf: Analysis of factors affecting players' performance and perception in multiplayer games. In: Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games, pp. 1-7 (2005)
15. M. Ries, P. Svoboda, M. Rupp: Empirical Study of Subjective Quality for Massive Multiplayer Games. In: Proceedigns of 15th Intl. Conf. Systems, Signals and Image Processing, pp. 181-184 (2008)
16. S. Choy, B. Wong, G. Simon, C. Rosenberg: A hybrid edge-cloud architecture for reducing on-demand gaming latency. Multimedia Systems 20(5), 503-519 (2014)
17. R. Shea, J. Liu, E. C. Ngai, Y. Cui: Cloud gaming: architecture and performance. IEEE Network 27(4), 16-21, 2013.
18. W. Cai, R. Shea, C. Huang, K. Chen, J. Liu, V. Leung, C. Hsu: A Survey on Cloud Gaming: Future of Computer Games. IEEE Access 4, 7605-7620 (2016)
19. H. Tian, D. Wu, J. He, Y. Xu, M. Chen: On Achieving Cost-Effective Adaptive Cloud Gaming in Geo-Distributed Data Centers. IEEE Trans. Circuits and Systems for Video Technology 25(12), 2064-2077 (2015)
20. Y. Lin, H. Shen, CloudFog: Leveraging Fog to Extend Cloud Gaming for Thin-Client MMOG with High Quality of Service. IEEE Trans. Parallel and Distributed Systems 28 (2), 431-445 (2017)
21. OpenNebula Systems: ONEedge.io: A Software-defined Edge Computing Solution. D2.1. Solution Framework – a. Technical Report (2020)
22. Federal Communications Commission: In the Matter of Preserving the Open Internet. Broadband Industry Practices, paragraph 20. GN Docket No. 09-191; WC Docket No. 07-52 (2010)
23. J. Bussiere, S. Zander: Enemy Territory Traffic Analysis. CAIA Technical Report 060203A, Swinburne University of Technology (2006)