End User-Managed Service Deployments in Microclouds at the Network Edge

Felix Freitag*, Leandro Navarro*, Mennan Selimi*†, Roger Pueyo Centelles*
* Universitat Politècnica de Catalunya, BarcelonaTech, Barcelona, Spain
† Max van der Stoel Institute, South East European University, North Macedonia

***Abstract*—Cloud computing is moving from data centers to the network edge. Edge computing introduces lightweight computing devices closer to where data are produced. Local processing enables faster response times for cloud-based services. In this approach, however, cloud elasticity still lies in the data center and not at the edge, and an edge device cannot run services locally beyond its capacity. In this paper we present microclouds as an end user-managed infrastructure built from low-cost home servers leveraging PaaS and SaaS at the network edge. We discuss the following aspects: 1) the microcloud platform architecture and the facilities it offers to the users to manage their services and applications, 2) support services to enable the microcloud provision, and 3) of a number of decentralized applications and their potential to support service provision in microclouds.**

***Index Terms*—edge cloud computing; decentralized computing infrastructures;**

I. Introduction

In this paper we present a practical implementation of the concept of an end user-managed edge cloud computing infrastructure by showing that, in home servers running the Cloudy platform [1], end users can deploy services and applications and share them with other users as a microcloud at the edge.

Commercial edge computing extends cloud computing services in data centers by means of additional edge devices located close to where the data are produced. Pre-processing in these devices allows to improve the response time of cloud-based applications by taking advantage of their proximity. In this architecture, data centers have control on the resource elasticity characteristic of cloud computing, but edge devices have a finite capacity and cannot elastically adapt to workloads which exceed their capacity.

In the microcloud approach, edge device elasticity is achieved by integrating distributed edge resources and services into a horizontal edge computing infrastructure [3]. Compared to data center-based edge computing, an increased level of resource elasticity at the network edge is achieved. Microclouds can be considered as a class of decentralized edge clouds [2].

In this paper we show how in these edge microclouds services and applications can be managed by end users. These microclouds are built with the Cloudy platform and they have been deployed in the Guifi.net [2] community network [4].

[1] http://cloudy.community/
[2] http://guifi.net/

II. Practical Microclouds

We demonstrate microclouds built from home servers by showing a microcloud deployment in the Guifi.net community network. Community networks are computer networks built by local communities in which individual citizens contribute off-the-shelf networking and computing devices. The networking capability of each device is shared, which allows to form a computer network which every member can use. Guifi.net is of one of the largest community networks in the world, with thousand of nodes and tens of thousands of users.

The networking hardware is often low-cost routers that interconnect different geographical locations. In addition, several participants of the community network run low energy consuming home servers (several examples are depictedd in Figure 1), which help to monitor the network and allow the participants to run applications –not only for personal use, but also to support the network management, and even to be shared with other participants.

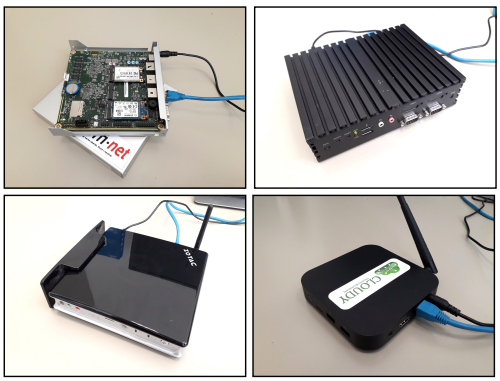

Fig. 1. Cloudy nodes built with different x86 embedded and mini-PCs.

These de-centrally managed and diverse home servers, when running the Cloudy software platform, are the target to be interconnected by software services to form a microcloud. Hence, the hardware devices in this microcloud are heterogeneous. They include, for instance, Raspberry Pis, x86 mini-PCs such as Minix nodes [3], and some desktop PCs.

The nodes of the microcloud are geographically distributed over the community network. Principally, nodes can be located

[3] http://minix.com.hk/products?category=9830

at the premises of the users/contributors. Nowadays, there are around 30 devices connected to this microcloud in Guifi.

III. Service and Application Management

Cloudy is meant to run on home servers. It integrates Serf [4], a gossip-based distributed software for exchanging messages, so that servers interconnect with each other to form a microcloud. In Linux distributions such as Ubuntu, Raspbian or Debian, end users can install the software for devices to become a Cloudy node by following the instructions given in the Cloudy repository. [4]

Once installed, users operate with Cloudy through a web GUI. Most services are provisioned as Docker containers. For service management, Cloudy provides several options: 1) predefined applications which users can run as Docker containers by clicking through Cloudy's web GUI, 2) personalized container deployment specifying in the *Docker FORM* menu a specific Docker image, 3) for applications consisting of several containers, Cloudy integrates *docker-compose* to parse configuration files.

Figure 2 shows a screenshot of the *Enterprise cloud* menu which gives access to the described container-based service deployment options. It is important to note that Cloudy allows a node owner to publish their deployed applications to other members of the Cloudy microcloud. Publication allows other users to be aware and use any shared service. Figure 2 illustrates a published Mosquitto broker service.

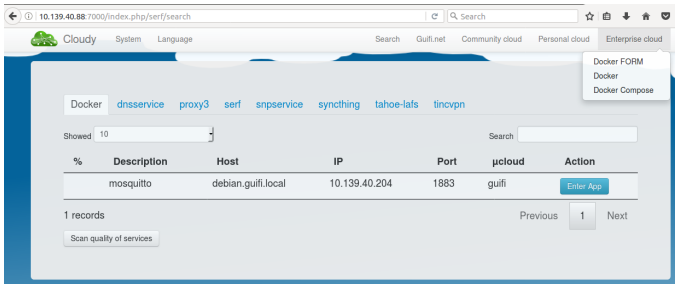

Fig. 2. A service published in the microcloud. The search service in Cloudy finds the Mosquitto broker deployed in a Cloudy node as Docker container.

IV. On-going Work and Outlook

Targets of our on-going IaaS/PaaS work include container orchestration and a consistent edge data storage layer.

Different to commercial edge computing platforms, Cloudy is conceived as an open platform, extensible with additional applications through container deployments, enabled by the local installation of Docker and *docker-compose* in Cloudy nodes. The graphical interface described before aims to make this task feasible to be carried out by end users without specific computer skills. However, a programmatic interface, equivalent to the Web UI, allows programmatic deployment in multiple services. That allows a resource or service manager process to control and manage the life-cycle of multiple containers running concurrently in a set of hosts to provide a given service [1]. Toward this end, we have explored the use of the Kubernetes orchestrator. One difficulty observed in the context of microclouds is the that each Cloudy node, being also used to run personal applications, retains its personal administrative domain. For orchestrating containers of an application over the worker nodes in a Kubernetes cluster, however, a single administrative domain seems to be expected.

It is important for the individual nodes of a distributed system to have a global and consistent view. For decentralized system such as a Cloudy microcloud, reasons include that such information could be used for local decisions and trigger accurate individual actions. Currently, for informing the nodes in the microcloud on new published services, Cloudy uses the Serf software, which applies gossip-based message dissemination to update the nodes in the network. While Serf is practically effective for the current requirements, the approach has theoretical limitations, e.g. lack of message delivery guarantees. We have integrated IPFS as a content-addressable, peer-to-peer method of storing and sharing data across nodes. We have deployed an eventually consistent distributed database named AntidoteDB [5] on Cloudy nodes. Its features on data consistency could fit to network partition and disconnection faced in edge microclouds. We have implemented a distributed monitoring application which leverages this database. Currently, we conduct a performance evaluations of the database on several Cloudy nodes.

V. Experimentation and Usage

Experimentation to gain a practical understanding of the presented microcloud can be done as follows:

1) Demo URL: We have provided a publicly accessible Cloudy instance to provide live access to the microcloud in Guifi.net through its Web GUI [6].
2) A Cloudy node can be easily built on an existing Linux distribution by a single script that installs all necessary packages [7].

Acknowledgment

This work was supported by the European H2020 framework programme project LightKone (H2020-732505), by the Spanish government contract TIN2016-77836-C2-2-R, and by the Catalan government contract AGAUR SGR 990.

References

[1] R. Baig, F. Freitag, and L. Navarro, "Cloudy in guifi.net: Establishing and sustaining a community cloud as open commons," *Future Generation Computer Systems*, 01/2018 2018.

[2] A. Chandra, J. Weissman, and B. Heintz, "Decentralized edge clouds," *IEEE Internet Computing*, vol. 17, no. 5, pp. 70–73, Sep. 2013. [Online]. Available: http://dx.doi.org/10.1109/MIC.2013.93

[3] Y. Elkhatib, B. Porter, H. B. Ribeiro, M. F. Zhani, J. Qadir, and E. Rivire, "On using micro-clouds to deliver the fog," *IEEE Internet Computing*, vol. 21, no. 2, pp. 8–15, Mar 2017.

[4] M. Selimi, A. M. Khan, E. Dimogerontakis, F. Freitag, and R. Pueyo Centelles, "Cloud services in the guifi.net community network," *Computer Networks*, vol. 93, Part 2, pp. 373 – 388, 2015, community Networks.

[4] https://github.com/Clommunity

[5] https://www.antidotedb.eu/

[6] http://demo.cloudy.clommunity/ (login: guest:guest)

[7] https://github.com/Clommunity/cloudynitzar