On Edge Microclouds To Provide Local Container-based Services

Roger Baig*, Roger Pueyo Centelles*, Felix Freitag†, Leandro Navarro†
*Fundació Privada per la Xarxa Lliure, Oberta i Neural Guifi.net. Mas l'Esperança, 08503 Gurb, Catalonia
{roger.baig, roger.pueyo}@guifi.net
†Department of Computer Architecture, Universitat Politècnica de Catalunya. Barcelona, Spain
{felix, leandro}@ac.upc.edu

***Abstract*—Edge computing has been proposed to enable more user-centric cloud-based services. Nowadays, edge computing is operational in industrial and consumer-oriented scenarios. An important limitation of today's solutions, however, is that the used hardware and software platforms are proprietary and closed, and cannot easily be leveraged to perform other services beyond the specific business case. For instance, the interaction among different edge platforms or service extension by third parties is in general not supported. As a consequence, the opportunity for local stakeholders to provide innovative tailored edge service with these platforms face important barriers. In this paper we present edge microclouds deployed on local servers and an implementation using containers for service provision. We show how the adopted container approach facilitates the users to create and share services at the network edge. With the presented approach, third parties can deploy more tailored and customized services at the network edge, enabling to better fulfill specific local needs and constraints. It is also an opportunity for building cloud-based service provision with a pool of local resource-constraint edge devices.**

***Index Terms*—edge cloud computing; community clouds;**

I. Introduction

The Fog [4][25] and Edge Computing [20] approaches extend the classic cloud architecture, which considers cloud computing services to run in large data centers. In edge cloud computing, by placing computing devices at the network edge, an additional infrastructure layer is added to the system model. As part of a cloud service, on these edge devices, specific functions are performed leveraging their closeness to the users or the data source.

Edge computing builds upon the advantages of cloud computing, but extends the traditional cloud services with the capacities of local processing. Edge computing solutions are already operational in many industrial and consumer-oriented scenarios, covering major application domains [13][15][12]. While we can observe large industrial players to contribute to open platforms [9], many of the solutions currently in the market are based on proprietary hardware and software platforms, making service extensions by external service providers and the interoperability with third-party services difficult.

Mobile cloud computing also integrates the edge computing approach [22]. In mobile cloud computing an additional infrastructure layer between the client and the data center cloud infrastructure is proposed, located at the base stations of the mobile phone operators. One important approach termed as *application offloading* has the purpose to off-load tasks and data from mobile applications to cloud infrastructures when the computing and memory demands cannot be met locally on the mobile device, leveraging also other wireless networks as alternative to the cellular network [14].

In this paper we consider a different edge computing model in which, by contrast to the above approaches, the users of edge services are enabled to collaborate and actively participate in the service provision, and contribute to sustain edge microclouds.

The cloud infrastructure of the presented case is located at the network edge and most devices that contribute to the resource pool are at the premises of the users or in installations of municipalities. The aim is a cloud which is formed by user-provided computing and communication resources to allow providing services of local interest. The software platform installed on these devices is open and can be extended with additional services by the participants. While the user can continue to access through the edge device the traditional cloud services, the collaboration and contributions among user devices enable a new horizontal layer of pooled edge resources and services, which we consider a *microcloud*.

The case we present is based on a real deployment of such microclouds in a collaborative community cloud in Guifi.net[1]. Guifi.net is a community network located in Spain, with a strong presence in the area of Barcelona. With more than 30.000 nodes and tens of thousands of users, it can be considered the largest community network worldwide. Microclouds have started to become operational in Guifi in 2015.

The paper is organized as follows: In the next section II we review several approaches for edge computing. Then in section III we position the concept of edge microclouds. We report in section IV the case study of a deployed microcloud in the Guifi community network. In Section V we show how local services are deployed. Section VI discusses some issues of these microclouds based on the experiences gained. Section VII summarizes our conclusions.

[1] https://guifi.net/

DOI: 10.1109/GIIS.2017.8169801

II. Edge Computing Strands

A. Centralized edge computing architectures

Edge and fog computing migrate cloud computing to the edge of the network, where lower-capacity nodes closer to the data source assist centralized cloud services [24]. These nodes at the network edge are part of a larger centralized cloud service. They can perform, for instance, initial data processing tasks for an Internet of Things (IoT) application [5]. Thus, we can consider that this model of centralized edge cloud services for processing data from edge sources consists in connecting, through the Internet, lightweight edge computing devices to centralized cloud data centers. In the data centers, the mass of data collected from IoT devices is processed and stored. This model is implemented by several big IoT players (Amazon[2], Microsoft[3], Dell[4], Intel[5], IBM[6], Redhat[7], etc.).

The vendor platforms used in this scenario are often rigid regarding their service provision flexibility, due to using closed platforms, which do not support extending the functions of the edge node by third-party services.

B. Container-based fog computing

Important initial works on fog computing were produced by networking hardware vendors [6]. Fog computing aims to be deployed on Customer-premises equipment (CPE), i.e. devices located in the households of the customers. Home routers and set-top-boxes were foreseen to carry other additional services beyond the basic network services, which telecom operators would then offer to end users. A study on how such advanced home gateways would extend home services was presented [26]. Users in this scenarios however act only as mere consumers, without any active participation.

A slightly more open approach is proposed in the fog computing scenario of Paradrop [27], where third-party repositories are introduced to download services from, and services are launched in Linux Containers (LXC), enabled in OpenWRT-based home routers.

In [17] container-based services are proposed for Single-Board-Computers used as gateways to conduct data processing at the edge. While with containers it is achieved to orchestrate services in a resource-constrained device, this work proposes a vertical architecture between the gateway and the data center cloud. There is no sharing of service *between* gateways.

C. Decentralized edge computing applications

Distributed services are a key enabler for decentralized platforms that exploit edge resources. A well know example are decentralized Peer-to-Peer (P2P) applications, which have demonstrated with millions of users the successful usage of edge resources. Almost always, however, P2P applications are very specific, e.g. file sharing, and cannot be adapted easily to other purposes, or extended to the provision of other services.

A generalization of P2P computing applications was suggested in [10]. In that work it was proposed within the context of edge computing that the users become part of edge cloud computing services and participate in the provided service.

D. Centralized distributed edge computing

Not only P2P file sharing, but also volunteer computing approaches have successfully exploited the idea of using distributed edge resources. In these approaches, a centralized component distributes work to voluntarily participating edge nodes. Early distributed voluntary computing initiatives were for instance SETI@Home [2] BOINC [1] and Folding@home [3]. These initiatives were driven by use cases which motivated the contributions of users, such as supporting the progress of biomedical and astrophysical research.

At the beginning of volunteer computing, desktop computers at the contributors' homes were the most used infrastructure. Volunteer computing however has evolved along with the evolution of hardware and technology, such that nowadays BOINC clients are available as smartphone apps, e.g. [23], and smartphone vendors have started supporting the idea of enabling users to contribute with mobile devices to the fields of medicine, science and environmental causes [11].

In volunteer computing platforms, the owner of the edge node need to install a client application to participate in the platform. We observe that edge nodes do not share services with each other but only with the central node. We also note that clients can perform the computation (often performing data analysis) only for the specific projects that they are subscribed to. Therefore, it is a use case which does not require from the software at the edge device any flexibility for performing other tasks.

E. Personal cloud services

Personal cloud services are another tool which enables end users to host services at the network edge. Several platform have appeared to support the creation of personal cloud services in different ways:

The Sandstorm tool [18] enables users to install local services from a kind of appstore with a few clicks. Services created by Sandstorm, however, target at personal use and are not meant to be shared with a community.

YunoHost [28] is a Debian-based distribution to facilitate end users the self-hosting of applications on local computers. The supported applications in YunoHost consist of a list of core applications and an extensible list of third-party provided applications. The platform mainly targets at providing personal cloud services deployed on single devices.

A data storage device is presented by Minebox [16]. Minebox is a hardware and software system for home users described as a data storage box to provide personal files against hardware failure, accidental deletion or cyberattacks.

[2] https://aws.amazon.com/iot

[3] https://azure.microsoft.com/en-us/documentation/articles/iot-suite-what-is-azure-iot/

[4] http://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Value_of_Analytics_at_the_Edge_Final.pdf

[5] http://www.intel.com/content/www/us/en/internet-of-things/white-papers/iot-platform-reference-architecture-paper.html

[6] http://www-03.ibm.com/software/products/en/internet-of-things-platform

[7] https://www.redhat.com/en/en/about/videos/intelligent-gateway

It leverages a peer-to-peer cloud storage network to which each Minebox becomes connected and which is secured by a blockchain. Minebox also offers a set of plugins for all major open source applications, which allows the user to extend a Minebox system with personal cloud services.

Nuvlabox [21] is another commercial product which develops the edge computing case targeting at personal services for home users. The Nuvlabox is a powerful small form factor PC to be installed at the users' homes to enable them performing local application deployments easily. It is connected to a remote data center from which it polls application images, which then are installed locally through an intuitive user interface.

III. The proposition of edge microclouds

While the above described systems and flexible container-based services in local computing devices can be considered state-of-the-art of personal cloud service provision, they can only perform until reaching the limit of the computing capacity of that gateway, but then will exhibit the problems of the centralized edge architectures needing to rely on the elasticity of *remote* data center resources to scale.

Different to the above described platforms, *edge microclouds* leverage the resources of other edge devices, and therefore can inherit the characteristics of cloud computing, like elastic resource and service availability, while microclouds are still located at the network edge.

Microclouds differ from the previously described edge cloud architectures regarding the pool of resources and services that are used. While, from a traditional cloud service provider perspective, edge computing infrastructure is mainly a resource that extends the capabilities of a data center based cloud service, in microclouds, the cloud resource pool for service provision are mainly the spare resources of distributed edge devices (even if it can extend towards data center cloud services to improve the edge service performance, e.g. for improved resilience). The collaborative element between users consists in resource donations, willingness to host distributed services, and active provision of services with the community of users in the microcloud.

Figure 1 illustrates the microcloud concept. To built these collaborative edge clouds, communication and computing devices at the network edge are leveraged. Different hardware device classes available at the network edge can become a microcloud resource. Mid-capacity servers may be found at municipalities and local acpSME. Low-capacity PCs or Single-board Computers (SBCs) are more and more used in domestic environments. High-end home gateways and routers to connect to the ISP can nowadays be chosen with important computing capabilities. Different to data center clouds, however, such edge microclouds will be distributed and heterogeneous in hardware, software, operational policies and administrative domains.

The services in these microclouds need to target the requirements of the community of the users they are to serve, so as to encourage contribution and participation by the users. Services enabled by these collaborative clouds should aim for new applications or customizations, and exploit the complementary element to traditional cloud services.

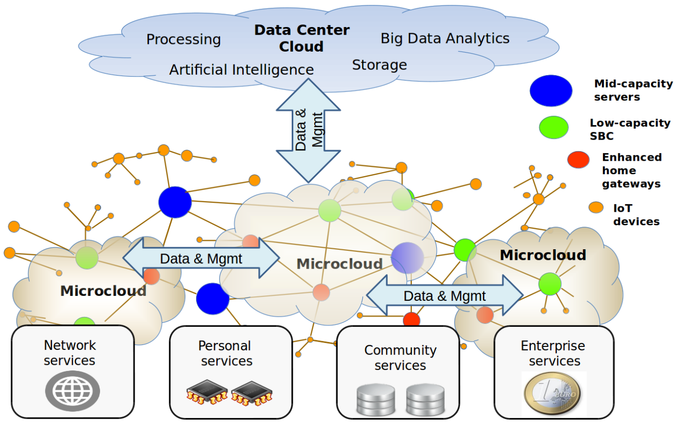


Fig. 1. Microclouds at the network edge.

IV. Microclouds in Guifi.net

A. Case study description

The edge microcloud we present as case a study is a community cloud deployed in the Guifi.net community network. Community networks are IP-based communication infrastructure which are built, owned and operated by local communities of citizens. While the original motivations of most community networks can be found in the need of their users to have Internet access, some community networks have also started to focus on the provision of local services within their network. Such a collaborative edge cloud service provision, which we can observe in community network clouds and report in this paper materializes microclouds, and is an alternative for how edge computing services can be created, managed and governed.

The Guifi.net community cloud consists of distributed heterogeneous computing devices contributed by the network participants. Often, these are inexpensive devices that can be classified as mini-PCs targeting home server applications. Such devices have low energy consumption to operate in a 24/7 mode.

The software system that runs on the devices is the Cloudy software [19], which is based on Debian Linux. Cloudy runs a set of common basic services provided in order to interconnect the devices and services forming part of the community cloud.

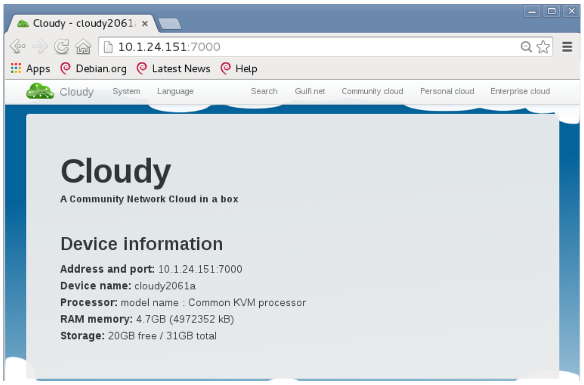
The devices of this edge community cloud are geographically distributed, since most of the contributed devices are located at the premises of the users (e.g. inside the home, garage, storage or service rooms in buildings). In addition, some (more powerful) hardware is located at municipality installations together with other city infrastructures (e.g. local data-centers, warehouses, street cabinets).

The Guifi.net community cloud has started to become operational in 2015. Nowadays, there are around 30 devices connected to the community cloud[8], and there are more than 50 users subscribed to the two mailing list, one for users[9] and the other for developers[10].

B. Microcloud software platform

Users of the Guifi.net community cloud are expected to install the Cloudy distribution on the contributed cloud nodes. Making Cloudy the default system for community network clouds ensures homogeneity in terms of a basic set of common services, which are needed for every participant to join and interact in the community cloud.

Figure 2 shows the web user interface after installation of Cloudy at the user's device. The Cloudy distribution provides a set of service categories, grouped as *Search*, network services of *Guifi.net*, *Community cloud*, *Personal cloud* and *Enterprise cloud*.



Cloudy

A Community Network Cloud in a box

Device information

Address and port: 10.1.24.151:7000
Device name: cloudy2061a
Processor: model name : Common KVM processor
RAM memory: 4.7GB (4972352 kB)
Storage: 20GB free / 31GB total

Fig. 2. Cloudy Web user interface.

In Cloudy we have recently switched to Docker as technology to enable the support for service personalization and customized service deployments. Docker has gained strong interest from research and industry, and major industries promote container technology for cloud computing [7].

Docker is based on the Linux container system, which is a lightweight virtualization technology. The virtualization layer considered is in the OS-Level, where the host kernel is shared between containers and the host system. This makes the processing of each container isolated from each other and achieves service performance as in bare-metal execution.

Moreover, it allows services to be easily shared and distributed across the network, with simple configuration commands and by building images of sets of applications, libraries and other files. Docker has created its own repository for user containers[11], the *Docker Hub*, allowing users to fetch images and build them on their own node.

C. Hardware for edge microclouds

Cloudy should be installable on any kind of domestic computing devices, which then can become part of the community network cloud. Figure 3 shows at the left side a typical node deployed in the Guifi community cloud. This device from Minix[12] comes with a low energy consuming Intel Z3735F (64-bit) processor, 2 GB of RAM and 32 GB of internal storage. Over the USB port, additional storage capacity can be added by the user. Cloudy has been tested and installed on desktop PCs, mini-PCs, as well as on low-resource single-board-computers (SBCs) such as RaspberryPi[13].

On the right side of Figure 3, we can see an example of a high-end home router[14], with not only connects the user to the optical fiber of the ISP, but also has computing capabilities at the level of SBCs. It is shipped with the OpenWRT operating system and is able to run services in LXCs.

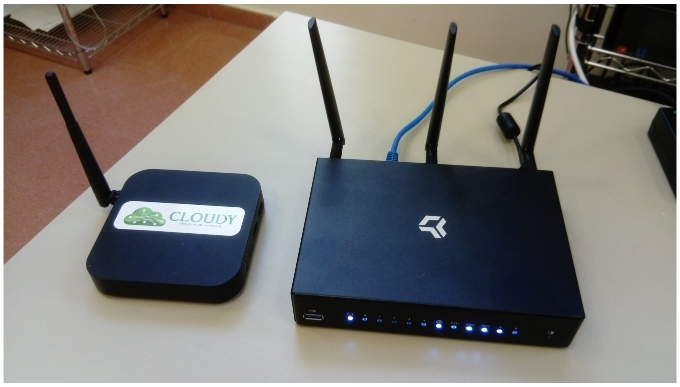


Fig. 3. Example of devices in the edge community cloud.

V. Local services in microclouds

A. Service provision management

Cloudy is operated by end users with different levels of technical skills. Therefore, an important requirement for Docker in Cloudy is that the integration of the control and configuration of Docker is such that the average end-user is able to install and run applications through Docker containers. The approach we followed is to hide the technical complexity of the Docker usage behind graphical web interfaces in Cloudy, more suitable for being managed by end users. In Cloudy, Docker is integrated in the GUI in the *Enterprise cloud* service category, where the user can enable (install) Docker images with a single click through the web browser.

B. Service deployment

In order to show the service deployment in this community cloud from the user perspective, we consider the scenario when a Cloudy user is willing to install and run an application in its Cloudy node and publish it to the other Cloudy users.

[8] The current number of Cloudy instances can be seen through a publicly available Cloudy instance at http://demo.cloudy.clommunity/, with login guest:guest.

[9] https://llistes.guifi.net/sympa/info/cloudy-users

[10] https://llistes.guifi.net/sympa/info/cloudy-dev

[11] https://hub.docker.com/explore/

[12] http://www.minix.com.hk/Products/MINIX-NEO-Z64-ANDROID-TV.html

[13] see boards and guides in http://wiki.clommunity-project/howto

[14] https://omnia.turris.cz/en/

The steps which the user has to conduct are installing and deploying the Docker container of the application in Cloudy and publishing this new application. Thus it will become known to the other Cloudy users. In the following we conduct the experiment to show these features.

The steps to deploy an existing container-based service is described for the case of the Kanban application. Kanban is a project management application which is available as a Docker image.

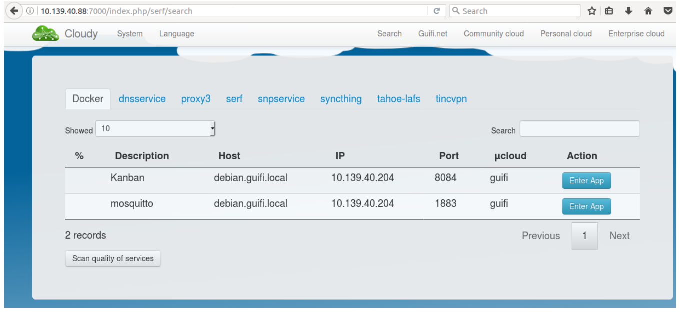
1) Select the application: In the *Docker* sub-menu under the *Enterprise cloud* tab in Cloudy, the user can find a list of pre-configured applications available that can be started as Docker containers. These applications can be installed and un-installed with a click on the web interface.
2) Deploy the application: One of the pre-installed services is Kanban, a web-based open-source project management application. Once the Cloudy user clicks on the *install* button, the container-based service is almost instantaneously created, and Kanban is ready to use, through its web interface, on a specific port.

C. Search for published services

The services deployed as Docker containers through the Cloudy web interface can be kept private for personal use or made public. In the latter case, they will be found by the search service running at other Cloudy instances. Publishing an application and, as a consequence, making it discoverable, is a first step to share an application among users.

When users want to make their application public, they activate this option by clicking the appropriate button in the *Publish* column. This step makes the Serf daemon in Cloudy to add the information about this new application in the regular exchange messages it gossips with other Cloudy nodes.

Figure 4 shows how a Cloudy user using the *Search service* finds the Mosquitto and Kanban applications deployed as Docker containers in the cloud, which are shared by their owners with the community.



Showed 10

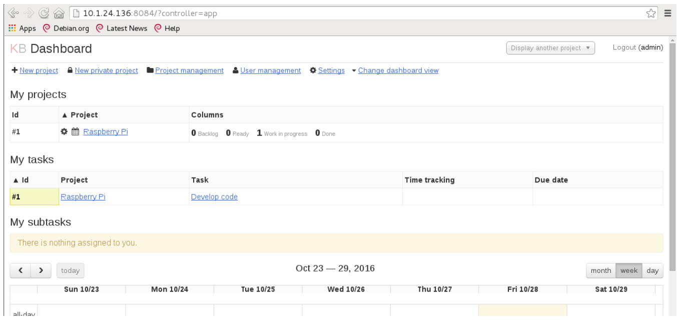
Search

%	Description	Host	IP	Port	µcloud	Action
	Kanban	debian.guifi.local	10.139.40.204	8084	guifi	Enter App
	mosquitto	debian.guifi.local	10.139.40.204	1883	guifi	Enter App

2 records

Fig. 4. A Cloudy user finds services publishes as Docker-based applications.

With the deployed Kanban application found, the user accesses the application by simply clicking on the *Enter App* button, which opens in the browser the Kanban login page. Figure 5 shows the accessed Kanban application which is ready to be used.



My projects

Id	▲ Project	Columns
#1	Raspberry Pi	0 Backlog 0 Ready 1 Work in progress 0 Done

My tasks

▲ Id	Project	Task	Time tracking	Due date
#1	Raspberry Pi	Develop code		

My subtasks

There is nothing assigned to you.

Oct 23 — 29, 2016

	Sun 10/23	Mon 10/24	Tue 10/25	Wed 10/26	Thu 10/27	Fri 10/28	Sat 10/29
all-day							

Fig. 5. Accessing the Kanban application deployed as Docker container in Cloudy.

VI. Discussion

Based on the results and experiences gained, we highlight in the following some issues to further address in order to consolidate the technical performance and sustainability of these edge microclouds.

A. Container-based service provision in microclouds

The usefulness of the services in microclouds is key for motivating users and encourage volunteer activity. While commercial players are increasing their service offer leveraging edge computing, microclouds need to differentiate by identifying services which benefit from the horizontal sharing of resources among users, a model not yet implemented by commercial providers.

Such services may include machine learning for data analytics, which nowadays is mostly done at cloud data centers. The increase of IoT data generation at the network edge, however, will also create the need to perform local analytics at the edge. Microclouds may serve as infrastructures to extend the capabilities of single edge devices to conduct such tasks. Security and privacy requirements for IoT will be use cases which can benefit from local data analytics support.

In the presented microclouds, Cloudy was used to deploy applications provided as single Docker containers. More complex applications, i.e. composed of a set of containers with dependencies among each other, should be supported next. In order to address these issues, we are in the process of integrating *Docker Compose* in Cloudy as the tool for deploying such applications.

B. Microcloud sustainability

Enabling commercial services is a promising approach to create an ecosystem around edge microclouds. The IoT domain has created many use cases that target to work with edge clouds [20]. To engage commercial players, the capability to deploy value-added services, often composed of several interacting components, is a requirement. Cloudy nodes participating in an edge microcloud infrastructure should therefore have the possibility to support commercial service deployments.

In the current Cloudy system, commercial applications would be deployed as Docker containers. The deployment is within a multi-tenant environment, where Quality of Experience (QoE) and Service Level Agreements (SLAs) for determined applications will need to be guaranteed. In [8] it is investigated how community gateway resources can be shared for private and community use. Further work in this line should be undertaken to understand the mechanisms needed for controlling the interplay of private, community and enterprise applications co-existing on edge devices.

VII. Conclusions

Edge microclouds are a decentralized form of edge computing which proposes an open flexible platform pooling together resources from the network edge for edge-based service provision. Edge device capabilities are extended by microclouds, providing thus the potential for scalable tailored services operated at the network edge.

The presented microclouds take advantage of the recent trend towards container-based service provision, and integrate this technology into the Cloudy platform. The integration of Docker in Cloudy, accessible through a web Graphical User Interface (GUI), has achieved hiding the technical complexity of container-based services and has made a step forward in bringing Docker usage to the reach of end users.

Suggested directions for future work include a deeper integration of Docker tools to enable complex application deployments. For the sustainability of microclouds, the definition of an ecosystem that coordinates volunteer contributions and commercial service provision is needed. While microclouds improve the resource elasticity of edge devices, optimization mechanisms will need to be developed to achieve QoE and SLAs in the ultimately resource-constrained microclouds.

Acknowledgment

This work was partially supported by the European Union's Horizon 2020 research and innovation programme under the project netCommons (H2020-688768), and the project FI-ESTA (H2020-643943), and by the Spanish government under contract TIN2016-77836-C2-2-R.

References

[1] D. P. Anderson, "BOINC : A System for Public-Resource Computing and Storage," in *5th IEEE/ACM International Workshop on Grid Computing*, Pittsburgh, USA, Nov. 2004, pp. 4–10. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1382809

[2] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, "SETI@home: an experiment in public-resource computing," *Communications of the ACM*, vol. 45, no. 11, pp. 56–61, Nov. 2002. [Online]. Available: http://portal.acm.org/citation.cfm?doid=581571.581573

[3] A. L. Beberg, D. L. Ensign, G. Jayachandran, S. Khaliq, and V. S. Pande, "Folding@home: Lessons From Eight Years of Volunteer Distributed Computing," in *8th IEEE International Workshop on High Performance Computational Biology (HiCOMB '09), within IPDPS*. Rome, Italy: IEEE, May 2009, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5160922

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12. New York, NY, USA: ACM, 2012, pp. 13–16. [Online]. Available: http://doi.acm.org/10.1145/2342509.2342513

[5] A. Chandra, J. Weissman, and B. Heintz, "Decentralized edge clouds," *IEEE Internet Computing*, vol. 17, no. 5, pp. 70–73, Sept 2013.

[6] "Internet of Things (IoT): Transform Data into Action at the Network Edge," Cisto Systems, Inc. [Online]. Available: http://www.cisco.com/web/solutions/trends/iot/fog-computing.html

[7] "Cloud native computing foundation (cncf)," Cloud Native Computing Foundation (CNCF), 2016. [Online]. Available: https://cncf.io/

[8] E. Dimogerontakis, R. Meseguer, L. Navarro, S. F. Ochoa, and L. Veiga, "Community sharing of spare network capacity," in *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, May 2017, pp. 299–304.

[9] "Open Source for IoT Eclipse IoT provides the technology needed to build IoT Devices, Gateways, and Cloud Platforms." The Eclipse Foundation. [Online]. Available: https://iot.eclipse.org/

[10] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, Sep. 2015. [Online]. Available: http://doi.acm.org/10.1145/2831347.2831354

[11] "HTC Power To Give," HTC Corporation. [Online]. Available: http://www.htc.com/us/go/power-to-give/

[12] "Watson Internet of Thi,ngs Platform," IBM. [Online]. Available: http://www-03.ibm.com/software/products/en/internet-of-things-platform

[13] "Intel® IoT Platform Reference Architecture," Intel Corporation. [Online]. Available: http://www.intel.com/content/www/us/en/internet-of-things/white-papers/iot-platform-reference-architecture-paper.html

[14] D. Liu, L. Khoukhi, and A. Hafid, "Data offloading in mobile cloud computing: A markov decision process approach," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[15] "Azure and Internet of Things," Microsoft Corporation. [Online]. Available: https://azure.microsoft.com/en-us/documentation/articles/iot-suite-what-is-azure-iot/

[16] "MINEBOX: Take control of your data and backups with a built-in blockchain and P2P network." Minebox IT Services GmbH. [Online]. Available: https://www.minebox.io/plugins/

[17] R. Morabito and N. Beijar, "Enabling data processing at the network edge through lightweight virtualization technologies," in *2016 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*, June 2016, pp. 1–6.

[18] "Sandstorm open source operating system for personal and private clouds," Sandstorm, 2016. [Online]. Available: https://sandstorm.io/

[19] M. Selimi, A. M. Khan, E. Dimogerontakis, F. Freitag, and R. P. Centelles, "Cloud services in the guifi.net community network," *Computer Networks*, vol. 93, Part 2, pp. 373 – 388, 2015, community Networks. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128615003175

[20] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, May 2016.

[21] "Nuvlabox Smart cloud-in-a-box appliance." SixSq. [Online]. Available: http://sixsq.com/products/nuvlabox/

[22] A. u. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 393–413, First 2014.

[23] "BOINC Android FAQ." University of California, Berkeley. [Online]. Available: http://boinc.berkeley.edu/wiki/Android_FAQ

[24] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Oct. 2014. [Online]. Available: http://doi.acm.org/10.1145/2677046.2677052

[25] B. Varghese, N. Wang, D. S. Nikolopoulos, and R. Buyya, "Feasibility of fog computing," *CoRR*, vol. abs/1701.05451, 2017. [Online]. Available: http://arxiv.org/abs/1701.05451

[26] J. Whiteaker, F. Schneider, R. Teixeira, C. Diot, A. Soule, F. Picconi, and M. May, "Expanding home services with advanced gateways," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 5, pp. 37–43, Sep. 2012. [Online]. Available: http://doi.acm.org/10.1145/2378956.2378962

[27] D. F. Willis, A. Dasgupta, and S. Banerjee, "Paradrop: A multi-tenant platform for dynamically installed third party services on home gateways," in *Proceedings of the 2014 ACM SIGCOMM Workshop on Distributed Cloud Computing*, ser. DCC '14. New York, NY, USA: ACM, 2014, pp. 43–44. [Online]. Available: http://doi.acm.org/10.1145/2627566.2627583

[28] "YunoHost: a server operating system aiming to make self-hosting accessible to everyone." YunoHost. [Online]. Available: https://yunohost.org/