CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 0000; **00**:1–15
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

Performance Evaluation of a Distributed Storage Service in Community Network Clouds

Mennan Selimi[1]*, Felix Freitag[1], Llorenç Cerdà-Alabern[1], Luís Veiga[2]

[1]*Department of Computer Architecture, Universitat Politècnica de Catalunya, Barcelona, Spain*
[2]*Instituto Superior Técnico (IST), INESC-ID Lisboa, Lisbon, Portugal*

SUMMARY

Community networks are decentralized communication networks built and operated by citizens, for citizens. The consolidation of todays cloud technologies offers now, for community networks, the possibility to collectively develop community clouds, building upon user-provided networks and extending toward cloud services. Cloud storage, and in particular secure and reliable cloud storage, could become a key community cloud service to enable end-user applications. In this paper, we evaluate in a real deployment the performance of Tahoe-LAFS, a decentralized storage system with provider-independent security that guarantees privacy to the users. We evaluate how the Tahoe-LAFS storage system performs when it is deployed over distributed community cloud nodes in a real community network. Furthermore, we evaluate Tahoe-LAFS in the Microsoft Azure commercial cloud platform, to compare and understand the impact of homogeneous network and hardware resources on the performance of the Tahoe-LAFS. We observed that the write operation of Tahoe-LAFS resulted in similar performance when using either the community network cloud or the commercial cloud. However, the read operation achieved better performance in the Azure cloud, where the reading from multiple nodes of Tahoe-LAFS benefited from the homogeneity of the network and nodes. Our results suggest that Tahoe-LAFS can run on community network clouds with suitable performance for the needed end-user experience.
Copyright © 0000 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: community networks; community cloud; cloud storage

1. INTRODUCTION

Wireless community networks [8] encompass an emergent model of infrastructure that aims to satisfy a community's demand for Internet access and information and communications technology (ICT) services. Most community networks originated in rural areas which commercial telecommunication operators left behind when deploying the broadband access infrastructure for urban areas. Different stakeholders of such a geographic area teamed up to invest, create, and run a community network as an open telecommunication infrastructure based on self-service and self-management by the users. Since the first community networks started more than ten years ago, they have become rather successful. Several large community networks in Europe have from 500 to 30000 nodes; these include Guifi.net [10], FunkFeuer [11], AWMN [12], and many more exist worldwide. Most of them are based on Wi-Fi technology (ad-hoc networks, IEEE 802.11a/b/g/n access points in the first hop, long-distance point-to-point Wi-Fi links for the trunk network), but

*Correspondence to: UPC-BarcelonaTech, Jordi Girona, 1-3, 08034, Barcelona, Spain
E-mail: mselimi@ac.upc.edu

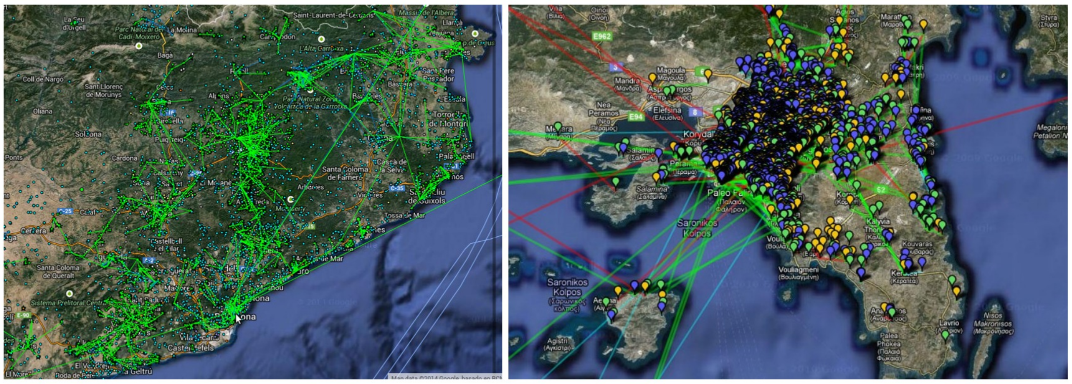


Figure 1. Guifi.net and AWMN nodes and links in the area around Barcelona and Athens.

also a growing number of optic fibre links have started to be deployed. Figure 1 shows as an example the wireless links and nodes of the Guifi.net and Athens Wireless Metropolitan Network (AWMN) community network in the area of Barcelona and Athens.

Community networks are a successful case of resource sharing among a collective, where resources shared are not only the networking hardware but also the time, effort, and knowledge contributed by its members that are required for maintaining the network. Resource sharing in community networks from the equipment perspective refers in practice to the sharing of the nodes network bandwidth. This sharing enables the traffic from other nodes to be routed over the nodes of different node owners. This is done in a reciprocal manner which allows community networks to successfully operate as IP networks. The sharing of other computing resources such as storage, which is now common practice in today's Internet through cloud computing, hardly exists in community networks, but it can be made possible through clouds in community networks, i.e. *community network clouds* [6].

The concept of community network clouds has been introduced in its generic form before, e.g.[1, 2], as a cloud deployment model in which a cloud infrastructure is built and provisioned for an exclusive use by a specific community of consumers with shared concerns and interests. Besides public, private, and hybrid models of cloud computing, a community cloud presents another option differing from the others in that it is designed with a specific community in mind, and with costs and responsibility shared among community members. Security concerns, government legislation, need for sophisticated control, or enhanced performance requirements in many scenarios make it difficult to rely on public clouds; and private clouds are not always the most cost-effective option.

We refer here to a specific kind of a community cloud in which sharing of computing resources is from within community networks [3, 5], using the application models of cloud computing in general. Establishing a community cloud involves many challenges, both in technological and socio-economic context, but also promises an interesting value proposition for communities in terms of local services and applications.

To conduct our evaluation of applications in a realistic scenario, our approach is to leverage on the cloud infrastructure provided by an ongoing cloud deployment in the Guifi.net community network [7] and a testbed deployed in community networks [8] [9].

The approach for performance assessment of applications in community networks which we propose in this paper is to set the experimental conditions as seen from the end user: experiment in production community networks, focus on metrics that are of interest to end users, and deploy applications on real nodes integrated in community networks.

We specifically look at a distributed secure storage application, Tahoe-LAFS [35] to be used within these wireless community-owned networks. After basic connectivity, storage is the most general and fundamental resource to support a distributed system, e.g. to store users' files, as in services like Dropbox, being fundamental for cloud take-up in community network scenarios. Tahoe-LAFS is an open-source distributed cloud storage system. Features of Tahoe-LAFS which are relevant for community networks are that it encrypts data at the client side, erasure codes it (data is broken into fragments, expanded, and encoded with redundant data pieces), and then disperses it among a set of storage nodes. This approach of Tahoe-LAFS results in high availability; e.g. even if some of the storage nodes are down or taken over by an attacker, the entire file system continues to function correctly, while preserving privacy and security. This is key for users' adoption. We host the Tahoe-LAFS distributed application on nodes of a community cloud spread inside the production community network. Furthermore, we compare the performance of Tahoe-LAFS when it is deployed in community clouds and in a commercial cloud such as Microsoft Azure.

The contributions of this paper are the following: 1) Verify the correct operation and adequate performance of the Tahoe-LAFS distributed file system in a real cloud system in a wireless community network. 2) Characterize the write and read performance of Tahoe-LAFS with different workloads generated by IOzone storage benchmark. 3) Present and show the feasibility of a case for storage sharing with performance evaluation conducted in a real setting of a deployed cloud system over a community network. We expect that these results will open the door to further application deployments and take-up of community clouds. The operational distributed file system which we investigated in this paper should encourage application developers to create innovative cloud-based community network services, and users to contribute resources which enable the hosting and usage of these services.

The rest of the paper is organized as follows. Section 2 defines the community clouds and discusses the requirements for such clouds. Section 3 explains the characteristics of the Tahoe-LAFS distributed file system that we use. Section 4 explains the experiment environment for community clouds and Azure cloud. Section 5 shows and discusses the results from our cloud deployment. Section 6 talks about related work, and section 7 concludes and discusses future research directions.

2. CLOUDS IN COMMUNITY NETWORKS

2.1. *Community Cloud Scenarios*

Our proposition is to deploy a distributed storage service in community networks using cloud-based resources. In this section we analyse the different components of this approach, which brought together, could enable the foreseen scenarios. We justify why the successful performance of such storage service, as reported in this paper, is key to motivate further steps. We therefore review first the conditions of community networks that must be considered when proposing scenarios and applications for community clouds. We then describe the scenarios we foresee and the objective of bringing applications into community networks.

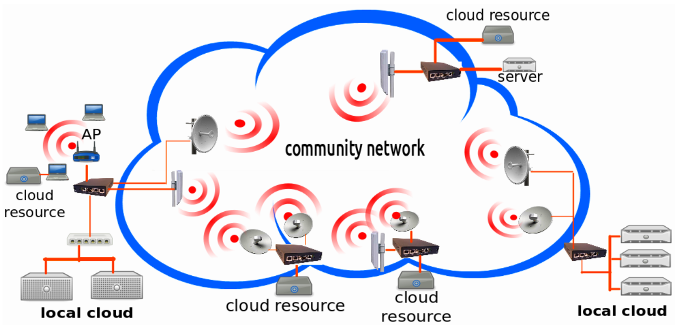


Figure 2. A Community Network with local clouds.

A community network typically has two kinds of nodes, i.e. super nodes and client nodes, and each type plays a different role in the network. For example, Guifi.net [10] includes these two main types of nodes, which are according to [14]:*terminal nodes* corresponding to end user nodes (client nodes), and *hubs* which serve traffic to end users (super nodes). Each terminal node has an unique connection to a hub that routes traffic, and hubs can have many connected terminal nodes.

We consider Figure 2 to derive the scenarios for clouds in community networks. The picture shows typical community nodes with a router and servers or clients attached to it. Some clients nodes (CN) are shown that are connected to the access point (AP) of a super node (SN). In addition, however, these community nodes have hosts, cloud resources, and even small local cloud data centers attached to them, illustrating the vision of the community cloud. So we define our *community cloud* as a cloud which is formed by small data centers spread in community networks (peer-to-peer network of nodes). Following a real ongoing deployment which we undertake in the Guifi community network [7], we identify the community cloud scenarios for these resources as follows.

Local Community Clouds: In the local community cloud, a super node is responsible for the management of a set of attached nodes that as hosts contribute cloud resources. From the perspective of the attached nodes, this super node acts as a centralized unit to manage the cloud services. The super node connects physically with other super nodes through wireless links, and logically in an overlay network to other local cloud managing super nodes. A local community cloud is not necessarily built with high-end machines; we rather see small data centres built with commodity desktop machines and managed by popular cloud operating systems such as Proxmox [30], OpenStack [31], or OpenNebula [32].

Federated Community Clouds: Given a set of local community clouds, multiple super nodes in a community network will connect and form federated clouds [15] [13]. Cloud federation enables the allocation of resources for a request on several local clouds. Distributed applications for community networks will need to be deployed over federated clouds [16] [17], similar to content-distribution networks, to attend local requests from remote community network nodes.

Micro-Clouds: The concept of micro-clouds is introduced to split the deployed community cloud nodes into different groups [4]. A micro-cloud refers to the cloud nodes which are within the same service announcement and discovery domain. Different criteria can be applied to determine to which micro-cloud a node belongs. Applying technical criteria (e.g. RTT, bandwidth, number of hops, resource characteristics) for micro-cloud assignment is a possibility to optimize the performance of several applications. In addition, social criteria may be used; e.g. bringing in a micro-cloud resources from users which are socially close may improve acceptance and the willingness to share resources and to maintain the infrastructure.

2.2. *Cloud-based Services in Community Networks*

To deploy applications and services in the community network, our approach is to provide a community network distribution, i.e. an operating system image which is prepared to be placed either as OS (operating system) of the community cloud node or in virtual machines of it. We have already developed such a distribution which we call *Cloudy*. Cloudy [38] is a Debian-based distribution, which has been equipped with a set of basic platform services and applications.

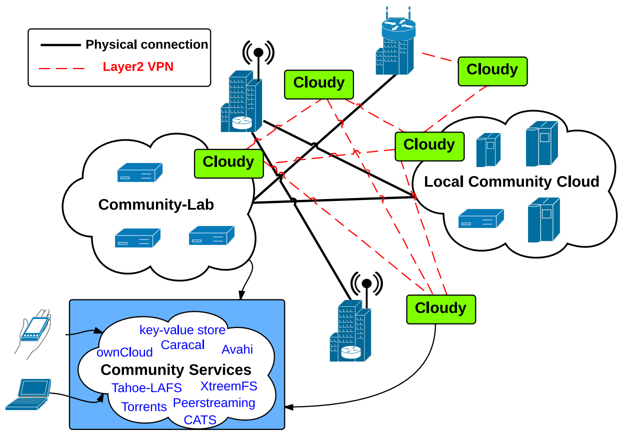


Figure 3. Cloud-based services in the distributed community cloud.

Figure 3 indicates some of the already integrated and foreseen applications of the Cloudy community cloud distribution. The Tahoe-LAFS distributed storage service, which we consider in this paper, and other services such as TincVPN [39], Serf [42], and Avahi [41] have already been added to the Cloudy distribution. The Serf and Avahi services are particularly important for the cloud system, because they allow the search for services, and with additional software even outside of the LAN. This way, services can be found from any node of the cloud.

Figure 3 also shows that the community services provided by the Cloudy distribution aim to run on different cloud resources provided by cloud management systems such as OpenStack or OpenNebula, and on other low-resource devices such as those provided by CONFINE Community-Lab testbed [8].

3. TAHOE-LAFS DISTRIBUTED FILESYSTEM

Tahoe-LAFS [35] is a decentralised storage system with provider-independent security. This feature means that the user is the only one who can view or modify disclosed data. The data and metadata in the cluster is distributed among servers using erasure coding and cryptography. The erasure coding parameters determine how many servers are used to store each file, which is denoted as N, and how many of them are necessary for the files to be available, denoted as K. The default parameters used in Tahoe-LAFS are K=3 and N=10 (3-of-10). The Tahoe-LAFS cluster consists of a set of storage nodes, client nodes, and a single coordinator node called the introducer. The storage nodes connect to the introducer and announce their presence, and the client nodes connect to the introducer to obtain the list of all connected storage nodes. The introducer does not transfer data between clients and storage nodes, but the transfer is done directly between them. The introducer is a first-point-of-contact for new clients or new storage peers, because they need it for joining the storage grid. When the client uploads a file to the storage cluster, a unique public/private key pair is generated for that file, and the file is encrypted (on the client side prior to upload), erasure coded, and distributed across storage nodes (with enough storage space) [23]. The location of erasure coded shares is decided by a server selection algorithm that hashes the private key of the file to generate a distinct server permutation. Then, servers without enough storage space are removed from the permutation, and the rest of the servers are contacted in sequence and asked to hold one share of the file. To download a file, the client asks all known storage nodes to list the number of shares of that file they hold, and in the subsequent rounds (second round-trip), the client chooses which share to request based on various heuristics such as latency, node load, etc. Figure 4 shows how an immutable file (created once and can be read repeatedly) is created. To create an immutable file, a client chooses a symmetric encryption key, uses that key to encrypt the file, and chooses erasure coding parameters K and N that erasure code the ciphertext into N shares and write each share to a different server.

Bringing Tahoe-LAFS Into Community-Lab: For our experiments we are using some nodes from the Community-Lab [29] testbed. The nodes run a custom OS (based on OpenWrt [33]) provided by the CONFINE project which allows running on one node several slivers simultaneously implemented as Linux containers (LXC) [40]. A sliver is defined as the partition of the resources of a node assigned to a specific slice (group of slivers). We can think of slivers as virtual machines inside a node. To deploy Tahoe-LAFS in the Community-Lab nodes, we use the Cloudy distribution, which contains Tahoe-LAFS, and place the introducer, storage, and client nodes of Tahoe-LAFS inside the slivers of the testbed. Figure 5 shows the resulting Tahoe-LAFS architecture used by our experiments in the Community-Lab testbed.

4. EXPERIMENTAL ASSESSMENT

Our objective is to evaluate the Tahoe-LAFS storage service in community network clouds and compare its performance with the performance obtained in a commercial cloud environment. The community network cloud is deployed in Guifi.net, and the commercial cloud used is Microsoft Azure cloud.

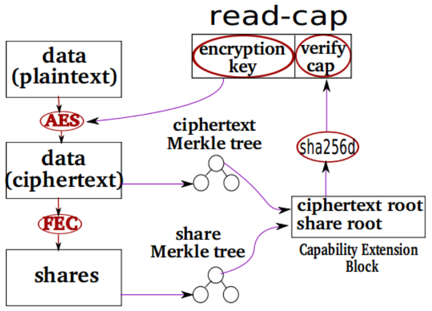


Figure 4. Immutable file creation in Tahoe-LAFS.

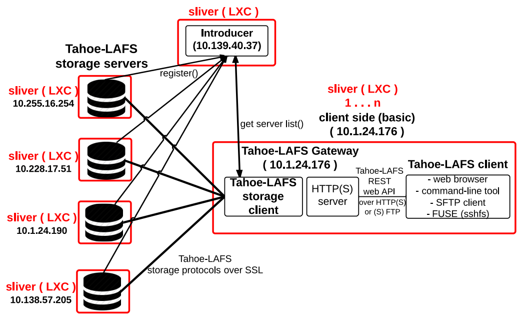

sliver (LXC)
Introducer (10.139.40.37)
Tahoe-LAFS storage servers
register()
sliver (LXC) 10.255.16.254
sliver (LXC) 10.228.17.51
sliver (LXC) 10.1.24.190
sliver (LXC) 10.138.57.205
get server list()
sliver (LXC) 1 . . . n client side (basic) (10.1.24.176)
Tahoe-LAFS Gateway (10.1.24.176)
Tahoe-LAFS storage client
HTTP(S) server
Tahoe-LAFS REST web API over HTTP(S) or (S) FTP
Tahoe-LAFS client
- web browser
- command-line tool
- SFTP client
- FUSE (sshfs)
Tahoe-LAFS storage protocols over SSL

Figure 5. Tahoe-LAFS deployed in the Community-Lab testbed.

4.1. Wireless Community Network Environment

We characterize wireless community networks by using experimental measurements in a production wireless mesh network. The network we consider, began deployment in 2009 in a quarter of the city of Barcelona, Spain, called Sants, as part of the *Quick Mesh Project* (QMP) [36]. In 2012, nodes from *Universitat Politècnica de Catalunya* (UPC) joined the network, supported by the EU CONFINE project [28]. We shall refer to this network as *QMPSU* (from Quick Mesh Project at Sants-UPC). QMPSU is part of a larger Community Network started in 2004 which has more than 30.000 operational nodes called Guifi.net. At the time of writing, QMPSU has around 54 nodes, 16 at UPC and 34 at Sants. There are two gateways, one in UPC Campus and another in Sants, that connect QMPSU to the rest of Guifi.net (Figure 6). A detailed description of QMPSU can be found in [18], and a live monitoring page updated hourly is available in the Internet [34].

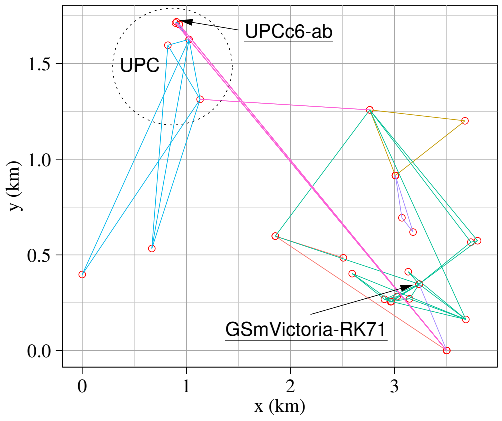


Figure 6. QMPSU topology. Gateways are underlined.

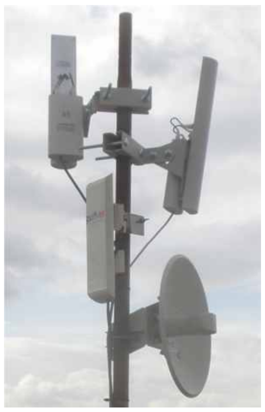


Figure 7. Outdoor routers.

Typically, QMPSU users have an outdoor router (OR) with a Wi-fi interface on the roof, connected through Ethernet to an indoor AP (access point) as a premises network. The most common OR in QMPSU is the NanoStation M5, which integrates a sectorial antenna with a router furnished with a wireless 802.11an interface. Some strategic locations have several NanoStations, that provide larger coverage (Figure 7). In addition, some links of several kilometers are set up with parabolic antennas (NanoBridges). ORs in QMPSU are flashed with the Linux distribution which was developed inside the QMP project. This distribution is a branch of OpenWRT [33] and uses BMX6 as the mesh routing protocol [25].

Measurements have been obtained by connecting via SSH to each QMPSU OR and running basic system commands. Connections have been made hourly during the entire month of *January, 2015*. Performance measurements of the throughput, RTT (round-trip time) to the Internet and between ORs, and number of hops between ORs have been taken. To limit the impact of the experiments on the users, measurements have not been done with all possible pairs of ORs, but only between each OR and its gateway. However, the measurements to the gateways can be used as an estimation of measurements between an arbitrary OR pair. The numbers of hops to the gateways have been derived from the routing tables, throughput using netperf, and RTT using pings.

To deploy Tahoe-LAFS in a realistic, community-cloud-like setting, we use the Community-Lab testbed, a distributed infrastructure provided by the CONFINE project, where researchers can deploy experimental services, perform experiments, or access open data traces. Some of the nodes of this testbed are connected directly to the outdoor routers (OR) of the QMP network. We use in total 10 of the nodes geographically distributed and connected to the 54 OR nodes. These nodes run a custom OS (based on OpenWRT), which allows running on one node several slivers simultaneously implemented as Linux containers. The slivers use the Cloudy operating system image which contains some of applications, e.g. Tahoe-LAFS, by default. The nodes in terms of hardware consist oj Jetway devices that are equipped with an Intel Atom N2600 CPU, 4 GB of RAM and 120 GB SSD.

4.1.1. Characterizing the Network Performance of Guifi.net. Figure 8 shows the average throughput and RTT to the gateway and the Internet, and the number of hops to the gateway obtained for every OR. The values are sorted by the throughput to the gateway. Standard deviation error bars are also shown. Internet values are measured using a server located outside of Gufi.net. Figure 8 reveals that the throughput to the Internet and the gateway are not linearly correlated. This is because the gateway located at UPC has a much better connection to the Internet. Thus, even if the throughput to the gateway is high, those nodes using the gateway in Sants have a low throughput to the Internet. Figure 8 demonstrates that the RTT has a stronger correlation with the number of hops than the throughput. Error bars in Figure 8 show that some nodes have an average number of hops with noticeable deviations. This variability has two causes: change in the routes, and selection of a different gateway.

Figure 9 shows the Empirical Cumulative Distribution Function (ECDF) of the average throughput to the gateway and the Internet (for the values depicted in Figure 8). On the top of the figure, the minimum/mean/maximum values are given. As shown in Figure 9, the overall mean throughput to the gateway is 17.4 Mbps with lower quartile of 4.7 Mbps, which reduces to a mean and lower quartile of 6.3 Mbps and 2.2 Mbps to the Internet. This reveals that the bottleneck is inside the portion of Guifi.net connecting QMPSU gateways with the Internet.

Similarly, Figures 10 and 11 present the ECDF of the average RTT to the gateway and the Internet, and number of hops to the gateway. Figure 10 indicates an overall average RTT to the gateway of 9.2 ms, with upper quartile of 14.3 ms, which increase respectively to 56.3 ms and 65 ms to the Internet. Finally, Figure 11 shows an overall average number of hops 2.53, with upper quartile of 3.4 hops.

We now investigate the impact of the traffic generated from the users on the network performance. Figure 12 illustrates the average traffic in both directions (referred to as *upload* and *download*) of the three busiest links over each hour of the day. These measurements have been obtained using the transmitted-bytes counters of the interfaces. Figure 13 shows the throughput averages in both directions on the same links as in Figure 12. These figures demonstrate a clear correlation between user traffic and link throughput. For instance, the upload link shown in the middle of Figure 13 is the most sensitive to user traffic. Figure 12 reveals that the average user traffic over the four lowest and highest busy hours of this link are 0.15 and 0.44 Mbps, respectively. Conversely, Figure 13 shows that the average throughput measured in this link during the same hours drops from around 32 to 24 Mbps.

4.2. Microsoft Azure Cloud Environment

The Microsoft Azure platform is the primary component of Microsoft cloud computing services. It relies on a global network of data centers managed by Microsoft to provide a collection of services that facilitate the development, deployment, and management of scalable cloud-based applications and services [43] [27]. Microsoft maintains data centers in five regions in North America, two

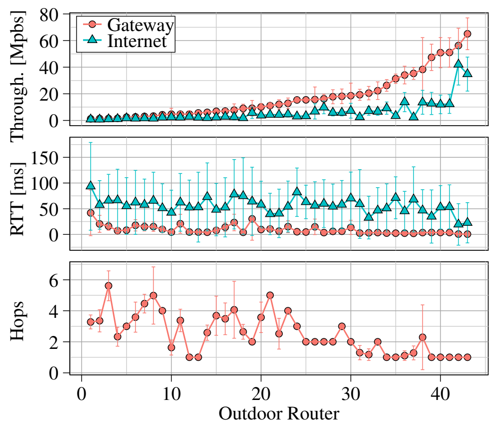


Figure 8. Average throughput and RTT to the gateway/Internet and number of hops to the gateway.

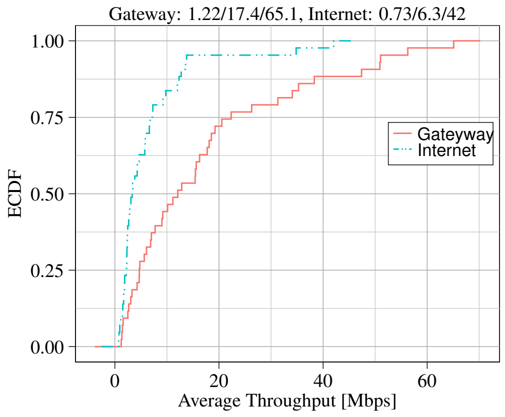


Figure 9. ECDF of the average throughput to the gateway/Internet.

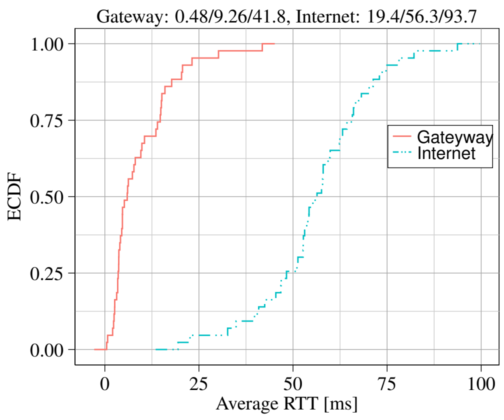


Figure 10. ECDF of the average RTT to the gateway/Internet.

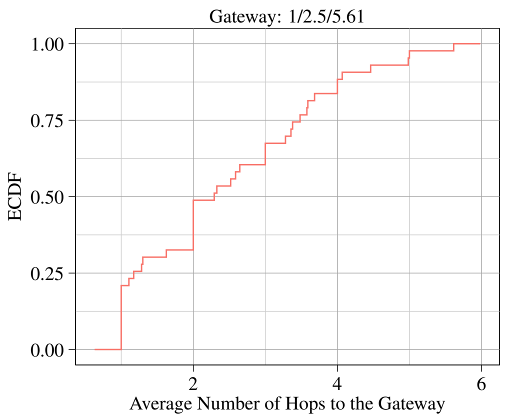


Figure 11. ECDF of the average number of hops to the gateway.

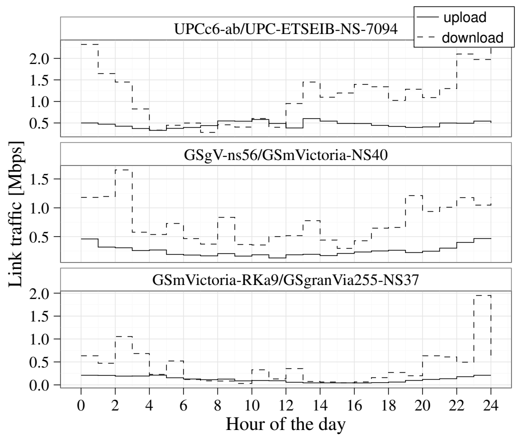


Figure 12. Average user traffic in the three busiest links.

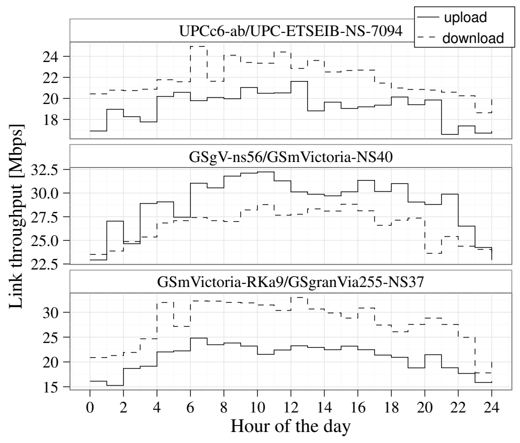


Figure 13. Average throughput in the three busiest links.

regions in Europe (West and East), two regions in Asia, and one region in Japan. Each region contains one or more data centers. In turn, each data center holds from ten to hundreds of thousands of servers. When we created the virtual machines in Azure cloud, we selected a data center region in West Europe. An alternative is to use the same service in multiple regions concurrently and direct users to the region closest to them. We created a Cloudy OS image and then used that image as a template to create a virtual machine managed by Microsoft Azure. Table I shows the Azure node characteristics and some network metrics obtained from them.

Table I. Azure nodes characteristics

Number of VMs	20
Size	A1(1 core, 1.75 GB memory)
Region	West Europe (WE)
Throughput between VMs	220-230 Mbits/sec
RTT between VMs	1-2 ms
Average throughput from the community network client	7-8 Mbits/sec

5. RESULTS

All tests were conducted using the IOzone cloud storage benchmark [37]. IOzone is a filesystem benchmark tool, which generates the cloud storage workload and measures various file operations. The benchmark tests file input/output (I/O) performance of many important storage-benchmarking operations, such as read, write, re-read, re-write, random read/write, etc. We run all 13 IOzone tests and vary the file size from 64 KB to 128 MB and we maintain a record length of 128 KB. An *-a* flag allows us to run all 13 tests. We add the *-b* flag to write the test output in binary format to a spreadsheet. We use a FUSE (Filesystem in Userspace) kernel module in combination with SSHFS (SSH Filesystem), an SFTP client that allows filesystem access via FUSE, to mount a Tahoe-LAFS directory to the local disk of the client. Tahoe's SFTP front end includes several workarounds and extensions to make it function correctly with SSHFS. When mounting with SSHFS, we disable the cache and use direct I/O and synchronous writes and reads, using the parameters *-o cache=no, big_writes, direct_io, and sshfs_sync*. We observe that the *-o big_writes* option to SSHFS improves write performance without affecting the read operations [26]. The distributed storage experiment is comprised of 15 runs of writing and reading files, where each run consists of 10 repetitions. Performance results presented in this paper are averaged over all the successful runs and are measured in MB/s referred to as operation speed. Tests with concurrent reading and writing were not conducted.

5.1. Community Networks Results

To better understand the impact that the network imposes on a community network environment, we established a Tahoe-LAFS cluster of 10 nodes that are geographically distributed and connected to the outdoor routers, as explained in section 4.1. Two sets of tests were conducted: one is when the reads and writes are initiated from a client which has the best connectivity in the network, such as best RTT to other nodes and best throughput (this is our baseline case). The other is when they are initiated from a client node which is the farthest node in the network (in terms of number of hops, RTT, and throughput to other nodes); this is referred to as the set 1 case in the graphs.

Figures 14 and 15 show the best and worst client write/read performance. Figure 16 depicts the summary of all tests performed with the IOzone benchmark. Median, first, and third quartile values are plotted for each data point. A few observations are noted below.

- In terms of network connectivity, both clients in community networks perform differently. This is related to the fact that the two clients are not connected in the same way to other nodes. The client in the baseline is better connected and is much closer in terms of RTT to the other nodes than the client in set 1.

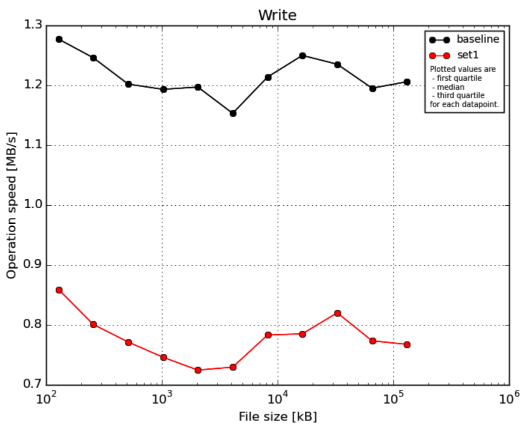


Figure 14. Performance of write operation in community networks.

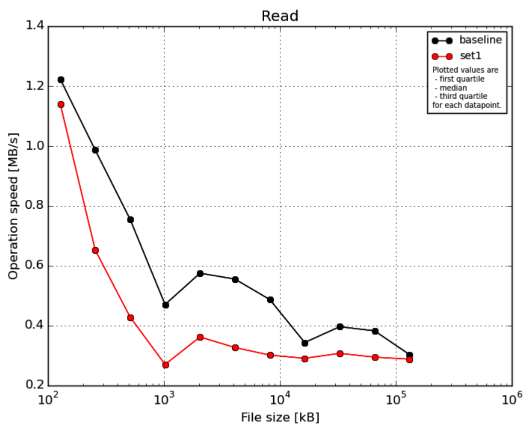


Figure 15. Performance of read operation in community networks.

- In terms of write performance, the baseline client performs better. Write performance for the baseline is higher and more stable than the read performance. As the file size increases, the write performance of the baseline client slightly decreases (minimum throughput achieved is 1.15 MB/s when writing a 4 MB file). The higher throughput is achieved (1.28 MB/s) when writing a small file (128 KB file). It is interesting to note that when writing smaller files, Tahoe-LAFS performs better. This is because the default stripe size of Tahoe-LAFS is well optimized for writing small objects (the stripe size determines the granularity at which data is being encrypted and erasure coded). The write performance of the set 1 client represents the average write performance of all community network nodes. The maximum write throughput achieved is 0.86 MB/s when writing a 128 KB file, and minimum write throughput is 0.72 MB/s when writing a 2 MB file. The throughput values achieved for the set1 client are in the range of measured values in Figure 9. Furthermore, write performance is affected by another factor; when writing new objects, Tahoe-LAFS generates a new public/private key, which is a computationally expensive operation.

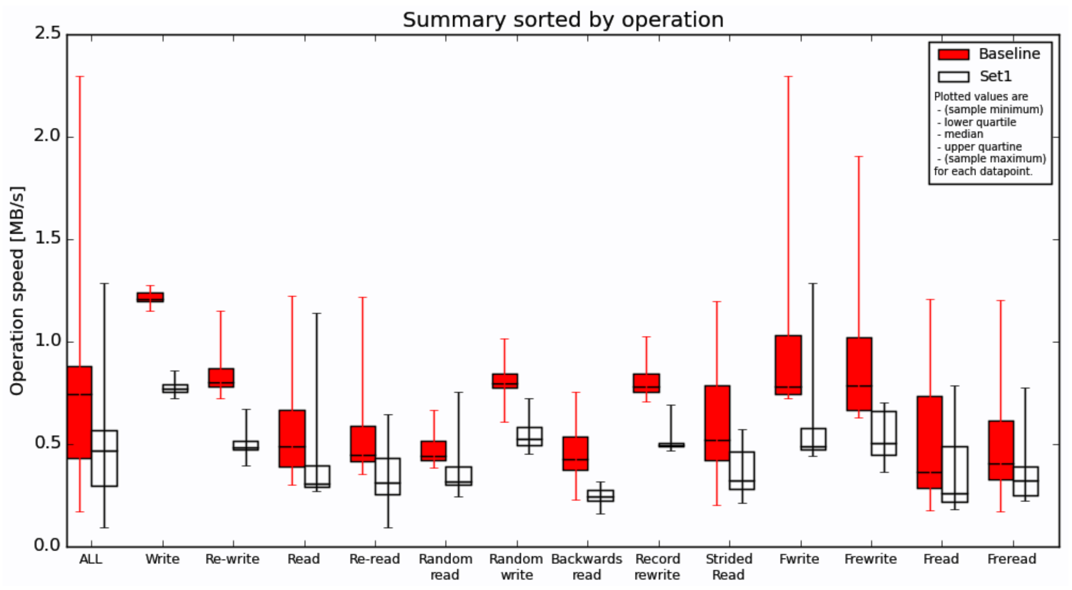
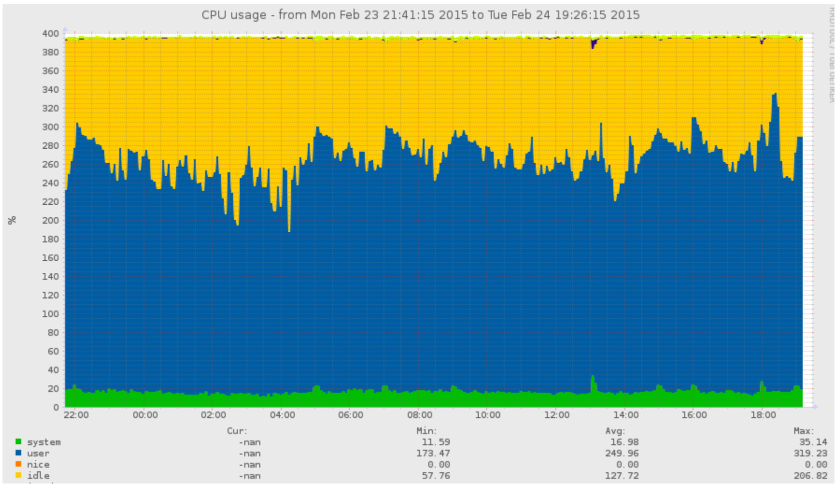


Figure 16. Summary of all storage benchmark operations for different tests in the Guifi.net community network.

- Read operations are accomplished by submitting the request to all storage nodes simultaneously; hence, the relevant peers are found with one round-trip to every node. The second round-trip occurs after choosing the peers from which to read the file. The intention of the second round-trip is to select which peers to read from, after the initial negotiation phase, based on certain heuristics. When reading from the storage nodes, the performance of both clients drops significantly because the file size increases, as shown in Figure 15. This is because when reading a file of 128 MB, a client must contact more Tahoe-LAFS storage peers to complete the shares of the file. In addition, reading the file system meta-object (i.e. the mutable directory objects) every time an object is accessed results in overhead, thus influencing the results.
- Figure 16 shows the summary of all tests performed with the IOzone benchmark. The benchmark tested file I/O performance for the 13 operations as presented in Figure 16. As indicated, the baseline client performs better than the set 1 client, reaching an average operation speed of 0.74 MB/s for all 13 tests performed.

The Figure 17 shows the CPU consumed from the Tahoe-LAFS client and Tinc overlay network [39] during the experiment hours. Tinc overlay network allows nodes to be connected to each other through VPN. Tinc Virtual Private Network (TincVPN) daemon uses tunneling and encryption to create a secure private network between cloud nodes, and this impacts the CPU performance of the node. Tahoe-LAFS uses the TincVPN to propagate the content to other nodes. Expensive and frequent hash read/write seeks that are needed to reconstruct shares on the storage peers and generation of a new public/private key every time a new object is created, can impact the CPU usage.



	Cur:	Min:	Avg:	Max:
🟩 system	-nan	11.59	16.98	35.14
🟦 user	-nan	173.47	249.96	319.23
🟧 nice	-nan	0.00	0.00	0.00
🟨 idle	-nan	57.76	127.72	206.82

Figure 17. CPU consumption of Tahoe-LAFS client. The blue color shows the CPU usage by Tahoe-LAFS and Tinc overlay network.

5.2. *Microsoft Azure Results*

Two sets of tests were conducted: one is when the reads/writes are initiated from a client node located in the same Azure cluster which is referred to as local read/write (baseline); the other is when they are initiated from a client node located in a different geo-location, which is referred to as remote read/write (set 1). The results are presented in Figure 18 and Figure 19. Figure 20 shows the summary of all tests performed with IOzone benchmark. In the results, throughput is given as a function of the file size. Median, first, and third quartile values are plotted for each data point. A few observations should be noted:

- In terms of local read/write performance (Figure 18 and Figure 19), Azure cloud system performs better. Write performance for the baseline is better and more stable than read

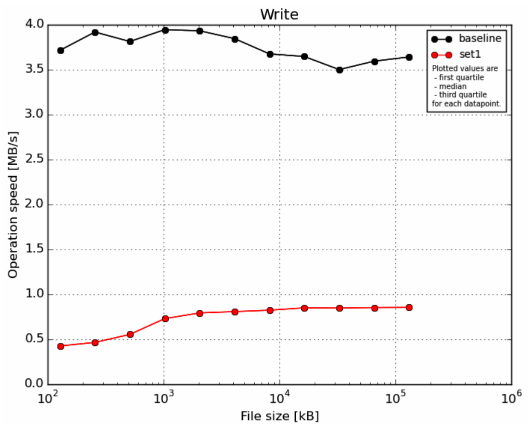


Figure 18. Performance of write operation in Azure cloud.

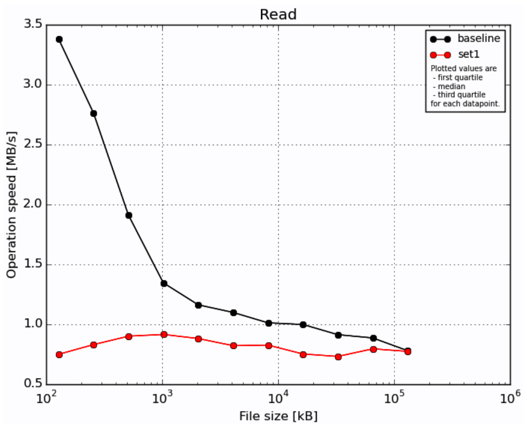


Figure 19. Performance of read operation in Azure cloud.

performance. As the size of the file increases, the read performance of the local client decreases by 0.7 MB/s when reading a 128 MB file. This is because when reading a bigger file (128 MB), a client has to contact more Tahoe-LAFS storage peers, to complete the shares of the file.

- When accessing the Azure cluster from a remote location (set 1 client), the performance drops significantly. The client is one of the community network nodes. The location of the Azure data center and a link from the client to the gateway can impact the results. The network from the client to the gateway (inside a community network) is a bottleneck.
- Figure 20 shows the summary of all tests performed with the IOzone benchmark in Microsoft Azure cloud. The IOzone benchmark tested file I/O performance for the 13 operations. As expected, the baseline client performs better than the set 1 client, reaching an average operation speed of 1.4 MB/s for all 13 tests performed. The average operation speed of the set 1 client for all tests is 0.7 MB/s.

6. RELATED WORK

In terms of providing cloud storage services with Tahoe-LAFS in WAN settings, Chen's paper [19] is the most relevant to our work. The authors deployed Tahoe-LAFS, QFS, and Swift in a multi-site environment and measured the impact of WAN characteristics on these storage systems. However, the authors deployed their experiments on a multi-site data center with very different characteristics to our scenario. Our approach is to assess the Tahoe-LAFS performance in the real context of community networks, and we use heterogeneous and less powerful machines as storage nodes.

The authors in [24] present a measurement study of a few Personal Cloud solutions such as DropBox, Box, and SugarSync. The authors examine central aspects of these Personal Cloud storage services to characterize their performance, with emphasis on the data transfers. They report that they found interesting insights such as the high variability in transfer performance depending on the geographic location; the type of traffic, namely inbound or outbound; the file size; and the hour of the day. Their findings regarding the impact of location on the performance is relevant for our work to better understand network dependence of distributed storage services.

Another work [20] implements a distributed file system for Apache Hadoop. The original Hadoop distributed file system is replaced with the Tahoe-LAFS cloud storage. The authors investigated the total transmission rate and download time with two different file sizes. Their experiment showed that the file system accomplishes a fault-tolerant cloud storage system even when parts of storage nodes had failed. However in the experiments only three storage nodes and one introducer node of

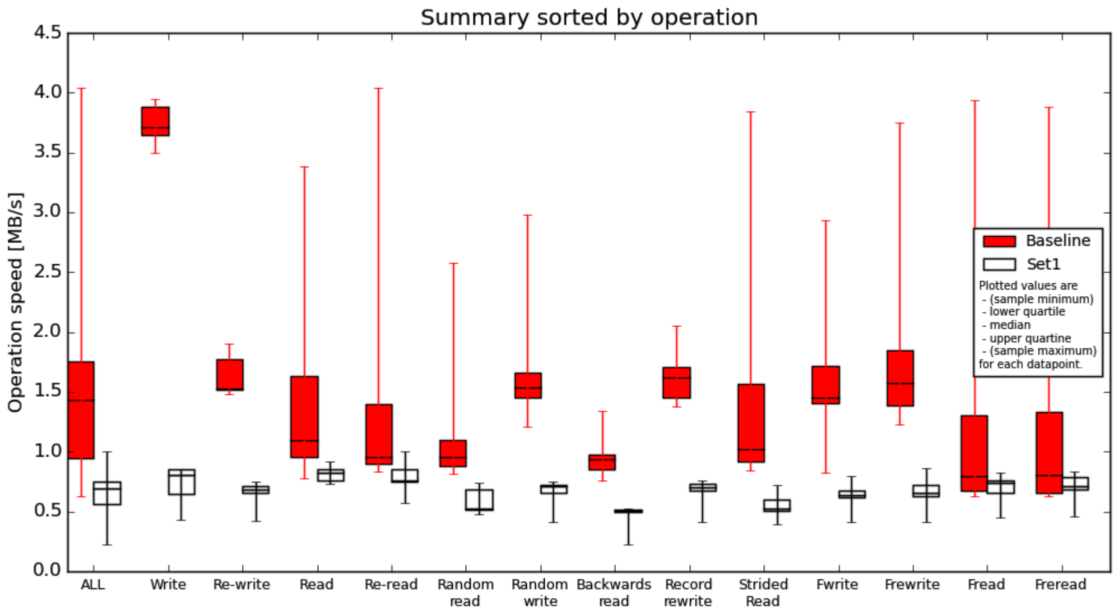


Figure 20. Summary of all benchmark operations for different tests in Azure cloud. The baseline refers to client located inside Azure cluster. Set 1 refers to client located outside Azure cluster (in a community network).

Tahoe-LAFS were used, and their experiments were run in a local context, which is an unrealistic setting for our scenario. Another paper [21] evaluates XtreemFS, Ceph, GlusterFS and SheepDog, using them as virtual disk image stores in a large-scale virtual machine hosting environment. The StarBED testbed with powerful machines is used for their experiments. Differently, we target a distributed and heterogeneous set of storage nodes.

The paper of Roman [22] evaluates the performance of XtreemFS under the IO load produced by enterprise applications. They suggest that XtreemFS has a good potential to support transactional IO load in distributed environments, demonstrating good performance of read operations and scalability in general. XtreemFS is an alternative candidate for implementing a storage service upon. Tahoe-LAFS, however, more strongly addresses fault-tolerance, privacy and security requirements.

From the review of the related work it can be seen that the experimental studies were not conducted in the context of community networks. In our work, we emphasized the usage of Tahoe-LAFS in a real deployment within community network clouds, to understand its performance and operational feasibility under real network conditions.

7. CONCLUSION AND OUTLOOK

Community networks would greatly benefit from the additional value of applications and services deployed inside the network through community clouds. The performance of such services running in community clouds and in comparison with running them in commercial clouds, however, was not yet shown by related work, but it is a needed step to encourage further application deployments in community clouds.

This paper evaluated the Tahoe-LAFS storage service performance on a community cloud and commercial cloud infrastructure. Tahoe-LAFS is a relevant application for community networks, because it offers privacy and security, as it encrypts data already on the client side, and it offers fault-tolerance regarding storage node failures due to erasure coding (replication factors). For the evaluation of Tahoe-LAFS, a real deployment in the community network cloud was conducted. Experiments assessed the read and write performance of the Tahoe-LAFS storage service in distributed and heterogeneous community network cloud nodes. In addition, Tahoe-LAFS was deployed in the Microsoft Azure commercial cloud, to compare and understand the impact of homogeneous network and hardware resources on the performance of the Tahoe-LAFS storage service.

We observed that the write operation of Tahoe-LAFS resulted in similar performance when using either the community network cloud or the Azure commercial cloud, but the read operation achieved better performance in the Azure cloud, where reading from multiple nodes of Tahoe-LAFS benefited from the homogeneity of the network and nodes. The results of our experiments suggest that Tahoe-LAFS can run on community network clouds with suitable performance for the needed end-user experience. Tahoe-LAFS could therefore be used as a back-end storage service for the operation of new application services using the resources of a community network.

Based on the observed successful performance and operation of the Tahoe-LAFS storage service in community network clouds, our experimental deployment will now become a permanent storage service open to real users. As the end-users start becoming more involved, we will be able to naturally extend our experiments with more storage nodes, more files coming from real data, and real usage. In addition, we anticipate that this storage service will effectively complement commercial offers and trigger innovative application in areas in which end-user involvement is required.

ACKNOWLEDGEMENTS

This work was supported by the European Framework Programme 7 FIRE Initiative projects CONFINE (FP7-288535) and CLOMMUNITY (FP7-317879), by the Universitat Politcnica de Catalunya BarcelonaTech and by the Spanish government under contract TIN2013-47245-C2-1-R. This work was supported by national funds through Fundação para a Ciência e a Tecnologia with reference UID/CEC/50021/2013.

REFERENCES

1. P. Mell and T. Grance, *The NIST Definition of Cloud Computing,* NIST Special Publication, vol. 800, no. 145, 2011.
2. A. Marinos and G. Briscoe, *Community Cloud Computing,* Cloud Computing, vol. 5931, pp. 472484, 2009.
3. A. M. Khan, U. C. Byksahin, and F. Freitag, *Prototyping Incentive based Resource Assignment for Clouds in Community Networks,* in 28th International Conference on Advanced Information Networking and Applications (AINA 2014). Victoria, Canada: IEEE, May 2014.
4. M. Selimi, F. Freitag, R. P. Centelles, A. Moll, L. Veiga, *TROBADOR: Service Discovery for Distributed Community Network Micro-clouds*, in: 29th IEEE International Conference on Advanced Information Networking and Applications (AINA15), 2015, pp. 642649.
5. J. Jimnez, R. Baig, F. Freitag, L. Navarro, and P. Escrich, *Deploying PaaS for Accelerating Cloud Uptake in the Guifi.net Community Network,* in International Workshop on the Future of PaaS 2014, within IEEE IC2E. Boston, Massachusetts, USA: IEEE, Mar. 2014.
6. M. Selimi, F. Freitag, R. Centelles, and A. Moll, Distributed Storage and Service Discovery for Heterogeneous Community Network Clouds, in 7th IEEE/ACM International Conference on Utility and Cloud Computing (UCC14). London, UK: IEEE/ACM, Dec. 2014.
7. J. Jimenez et al., *Supporting Cloud Deployment in the Guifi.net Community Network,* in 5th Global Information Infrastructure and Networking Symposium (GIIS 2013), Trento, Italy, Oct. 2013.
8. B. Braem, C. Blondia, C. Barz, H. Rogge, F. Freitag, L. Navarro, J. Bonicioli, S. Papathanasiou, P. Escrich, R. Baig Vias, A. L. Kaplan, A. Neumann, I. Vilata i Balaguer, B. Tatum, and M. Matson, *A case for research with and on community networks,* SIGCOMM Comput. Commun. Rev., vol. 43, no. 3, pp. 6873, Jul. 2013.
9. M. Selimi, F. Freitag, R. Pueyo, P. Escrich, D. Marti, and R. Baig, *Experiences with Distributed Heterogeneous Clouds over Community Networks,* in SIGCOMM Workshop on Distributed Cloud Computing (DCC14), within ACM SIGCOMM. Chicago, USA: ACM, Aug. 2014.
10. Guifi.net: *Open, Free and Neutral Network Internet for everybody.* [Online]. Available: http://guifi.net.
11. Funkfeuer: *Free Net* [Online]. Available: http://www.funkfeuer.at/.
12. AWMN: *Athens Wireless Metropolitan Network* [Online]. Available: http://www.awmn.net/content.php.
13. *EGI: European Grid Infrastructure, Federated Cloud* [Online]. Available: https://www.egi.eu/infrastructure/cloud/.
14. D. Vega, L. Cerdà-Alabern, L. Navarro, and R. Meseguer, *Topology patterns of a community network: Guifi.net,* in 1st International Workshop on Community Networks and Bottom-up-Broadband (CNBuB 2012), within IEEE WiMob, Barcelona, Spain, Oct. 2012, pp. 612619.
15. R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente, *IaaS Cloud Architecture: From Virtualized Datacenters to Federated Cloud Infrastructures,* Computer, vol. 45, no. 12, pp. 6572, Dec. 2012.
16. R. Buyya, R. Ranjan, and R. N. Calheiros, *InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services,* Algorithms and Architectures for Parallel Processing, vol.6081, pp. 2031, Mar. 2010.

17. M. Gall, A. Schneider, and N. Fallenbeck, *An Architecture for Community Clouds Using Concepts of the Intercloud,* in 27th International Conference on Advanced Information Networking and Applications (AINA13). Barcelona, Spain: IEEE, Mar. 2013, pp.7481.
18. L. Cerdà-Alabern, A. Neumann, and P. Escrich, *Experimental evaluation of a wireless community mesh network,* in Proceedings of the 16th ACM International Conference on Modeling, Analysis, Simulation of Wireless and Mobile Systems, ser. MSWiM 13. New York, NY, USA: ACM, 2013, pp. 2330.
19. Y.F. Chen, S. Daniels, M. Hadjieleftheriou, P. Liu, C. Tian, and V. Vaishampayan, *Distributed storage evaluation on a three-wide interdata center deployment*, in Big Data, 2013 IEEE International Conference, 2013, pp. 1722.
20. F.H. Tseng, C.Y. Chen, L.D. Chou, and H.C. Chao, *Implement a reliable and secure cloud distributed file system,* in Intelligent Signal Processing and Communications Systems (ISPACS), 2012 International Symposium on, 2012, pp. 227232.
21. K. Shima and N. Dang, *Indexes for Distributed File/Storage Systems as a Large Scale Virtual Machine Disk Image Storage in a Wide Area Network.*.
22. R. Talyansky, A. Hohl, B. Scheuermann, B. Kolbeck, and E. Focht, *Towards transactional load over xtreemfs,* CoRR, vol. abs/1001.2931, 2010.
23. Z. Wilcox-OHearn and B. Warner, *Tahoe: The least-authority filesystem* in Proceedings of the 4th ACM International Workshop on Storage Security and Survivability, ser. StorageSS08, New York, NY, USA:ACM, 2008, pp. 2126.
24. Gracia-Tinedo, R.; Sanchez Artigas, M.; Moreno-Martinez, A.; Cotes, C.; Garcia Lopez, P., *Actively Measuring Personal Cloud Storage* 2013 IEEE Sixth International Conference on Cloud Computing (CLOUD),vol., no., pp.301,308, June 28 2013-July 3 2013, doi: 10.1109/CLOUD.2013.25
25. A. Neumann, E. Loopez, and L. Navarro. *An evaluation of bmx6 for community wireless networks.* In 1st International Workshop on Community Networks and Bottom-up-Broadband(CNBuB'2012), pages 651658, Barcelona, Spain, Oct. 2012.
26. Aditya Rajgarhia and Ashish Gehani. 2010. *Performance and extension of user space file systems*. In Proceedings of the 2010 ACM Symposium on Applied Computing, SAC'10.
27. B.Calder et al. *Windows Azure Storage: a highly available cloud storage service with strong consistency*. In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles (SOSP'11). ACM, New York, NY, USA, 143-157. DOI=10.1145/2043556.2043571
28. CONFINE Project: *Community Networks Testbed for the Future Internet* http://confine-project.eu/ , FP7 European Project 288535.
29. Community-Lab: *Community Networks Testbed by the CONFINE project* http://community-lab.net/
30. Proxmox: *Virtual Environment* https://www.proxmox.com/en/
31. OpenStack: *The Open Source Cloud Operating System* https://www.openstack.org/
32. OpenNebula: *Cloud Management Platform* http://opennebula.org/
33. OpenWrt: *Linux distribution for embedded devices.* https://openwrt.org.
34. QMPSU: *Sants-UPC monitoring web page.* http://dsg.ac.upc.edu/qmpsu.
35. Tahoe-LAFS: *The Least-Authority File Store* https://www.tahoe-lafs.org/trac/tahoe-lafs
36. qMP: *Quick Mesh Project* http://qmp.cat/.
37. IOzone: *Filesystem Benchmark* http://www.iozone.org/.
38. Cloudy: *Community Network distribution based on Debian GNU/Linux* http://cloudy.community/.
39. TincVPN: *Tinc Virtual Private Network* http://www.tinc-vpn.org/.
40. LXC: *Linux Containers* https://linuxcontainers.org/
41. Avahi: *Service Discovery* http://avahi.org/.
42. *Serf* https://www.serfdom.io/.
43. Azure: *Microsoft's cloud platform* http://azure.microsoft.com/en-us/overview/what-is-azure/