Project Title: Clommunity: A Community networking Cloud in a box

Deliverable Title: Holistic Abstract Network and Service Architecture

Deliverable number: D3.4

Version 1.0

This project has received funding from the European Union's Seventh Programme for research, technological development and demonstration under grant agreement No 317879

Project Acronym: CLOMMUNITY
Project Full Title: A Community networking Cloud in a box.
Type of contract: Collaborative project (STREP)
contract No: FP7-ICT-317879
Project URL: http://clommunity-project.eu/

Editors:	Hooman Peiro Sajjad (KTH), Amin Khan (UPC), Felix Freitag (UPC), Vladimir Vlassov (KTH)
Deliverable nature:	Report (R)
Dissemination level:	Public (PU)
Contractual Delivery Date:	30/06/2015
Actual Delivery Date:	30/06/2015
Suggested Readers:	Project partners
Number of pages:	26
Keywords:	WP3, Community clouds, architecture, resource management, self-management, scalable service overlays
Authors:	Vladimir Vlassov (KTH), Hooman Peiro Sajjad (KTH), Paris Carbone (KTH), Ying Liu (KTH), Jim Dowling (SICS), Lars Kroll (KTH, SICS), Alexandru - Adrian Ormenisan (KTH, SICS), Amin Khan (UPC), Mennan Selimi (UPC), Felix Freitag (UPC), Leandro Navarro (UPC), Davide Vega (UPC)
Peer review:	Roc Meseguer (UPC)

Abstract

This document presents the work carried out in WP3 during the last reporting period of the CLOMMUNITY project, extending previous work reported in D3.1, to discuss the revised and final version of the network and service architecture for community clouds.

Executive Summary

In this document, our revised version of the community networking Clouds' network and service architecture are presented.

Based on the community network context, the requirements for a Community Cloud are defined. A multi-layer architecture is proposed to satisfy those requirements. The architecture consists of five layers: hardware, core, middleware, services and front end. Resources shared by the community members and connected through the community network are in the hardware layer. A resource can be either bare metal or virtual machine. On top of that Community Cloud distribution, Cloudy, provides community Cloud services by connecting to other Cloudy instances. In the middleware layer, there are network coordination and supporting services and tools to provide connectivity. On the service layer, there is a service-discovery service that enables users to discover and access available Cloud services. Services such as Tahoe file system and GVoD video-on-demand are placed in this layer. Front end layer provides a user interface to enable users to configure and add services.

Contents

List of Figures

1 Introduction

1.1 Contents of the deliverable

This deliverable reports the work carried out in WP3 regarding the architecture of the Community Cloud system in final period of the CLOMMUNITY project to summarize the architectural considerations and our proposed solutions, by extending D3.1 reported at M06. We discuss the work carried out in task T3.1, regarding the network and service architecture and the motivations behind our proposed solutions.

1.2 Relationship to other CLOMMUNITY deliverables

The deliverable D3.4 is the final revised version of D3.1 delivered in M06. D3.1 contains the requirements for the network and service architecture, and presented already a first version of the architecture itself. Building upon D3.1, we discuss and extend the final network and service architecture, including a layered architecture and its implementation, based on the insights we have obtained from results of WP4 and WP2.

1.3 Research challenges addressed

Our goal is a community cloud system that is deployed in the real-world settings of the community networks, therefore it is a must that the architecture of such a community cloud adapts to the socio-technical characteristics and the particular requirements of the community networks. As a result, we study the social and economic nature of community networks, and their technical and technological aspects, e.g. sharing of resources, social and topological aspects, and this motivates our design for the holistic architecture of a community cloud system.

A community cloud is a combination of a number of cloud systems being run and managed independently by the different community members. The community cloud bridges in different aspects the gap between the public cloud, the general purpose cloud available to everyone, and the private cloud, available to only a limited set of users with user-specific services. Community Cloud has a set of design considerations that need to be satisfied in order to be deployed and adopted successfully by the community. Autonomy, security, self-management, ease of use are some of those considerations.

Based on the specific design considerations of the Community Cloud system, we foresee realising the community cloud by deploying a community cloud platform tailored to the specific infrastructure and context of community networks. In our effort, we focus on providing a framework that would allow users to share resources and access collaboratively-built services. We propose a framework that can serve as the core of a community cloud system. Our community cloud framework is a distributed bottom-up resource sharing and collaborative services platform.

Based on the proposed architecture, we implemented Cloudy, which is a distribution based on Debian GNU/Linux, aimed at end users, to foster the transition and adoption of the community network cloud environment. A Cloudy instance can be run directly on a bare metal machine or on a virtual machine. Independently of the hardware that Cloudy runs on, connectivity to other Cloudy instances is obligatory in order to fully exploit the potential of Cloudy. Cloudy provides some services to support cloud-based services in community networks. It provides virtualization, service discovery and network coordination services. It also provides user services such as video-on-demand and storage. Cloudy provides an interface that enables users to configure and access the services.

2 Community Networks' Socio-Technical Context

Our goal is a community cloud system that is deployed in the real-world settings of the community networks, so it is a must that the architecture of such a community cloud adapts to the socio-technical characteristics and the particular requirements of the community networks. In this section, we study the social and economic nature of community networks, and their technical and technological aspects, and this motivates our design for the holistic architecture of a community cloud system.

2.1 Sharing of resources

Community networks are a successful case of resource sharing among a collective. The resources shared include networking, hardware but also each community network participant's time he/she donates, in a different extent, for maintaining the network. Computing and storage resource sharing through networks, such as is common practice nowadays through Cloud Computing, however is lacking in community networks. Thus, any service offered in community networks runs on machines exclusively dedicated to a single member.

2.1.1 Shared bandwidth

Resource sharing in community networks from the equipment perspective refers in practice to the sharing of the nodes' bandwidth. This sharing enables that traffic from other nodes is 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.

2.1.2 Shared Internet access with federated proxies

Internet access in the Guifi community network is offered through a group of federated proxies. There are currently more than hundred Internet proxies declared in Guifi.net. Proxies are hosted and maintained voluntarily by various individuals. In Guifi.net, a registered member can gain access to an Internet proxy by requesting being part of the group of users that can gain Internet access. A separate user account is being issued for accessing those proxies and is explicitly used for Internet access. A registered proxy user can have access to any of the federated proxies in Guifi.net. Furthermore, user account information is maintained via the Lightweight Directory Access Protocol (LDAP).

2.1.3 Shared time and knowledge

The community network's operation and maintenance is the product of the aggregated effort of its individual members in the same way as the network infrastructure is the aggregated wireless equipment contributed by its members. Individuals effort translates into time and knowledge spent to configure and further improve network operability. In such a decentralized infrastructure, community network individuals are responsible for their own network devices on which they have full control. Therefore, the community network as a whole achieves a level of sustainability via the collaborated effort of the member contributions.

2.1.4 Crowd funding

Typically, each node of a community network belongs to one owner (in most cases it's the person or organization who bought it and on whose premises the node is located). However, there are nodes in Guifi.net that have been crowd funded instead. Crowd funding of nodes has been adopted in various cases due to necessary infrastructure needs over groups of people. For example, users of isolated geographical zones of Guifi.net co-established super nodes, which will be explained below, to gain connectivity to other zones. In such cases, the overall cost of a node setup was shared over multiple individuals. The location of such crowd funded nodes often follows strategic considerations, aiming to improve or optimise marginal resources in terms of performance or connectivity with the help of the added infrastructure. As a concept, crowd funding is yet another example of a collaborative contribution towards marginal benefits.

2.2 Social aspects

As a means to deal with physical constraints Guifi.net is organized into zones. Practically, a zone corresponds to a spatial topology such as a village, a small city, a region, or district of a larger city. Each zone might consist of one to an arbitrary number of geographically related interconnected nodes. From a social perspective, there are two types of user communities, or social networks formed throughout Guifi.net, each maintaining its own mailing list. First, there exists a global network community where general technical and organisational issues are resolved. On that global level there are participant members from the whole Guifi.net network in addition to external participants who have an interest and motivation to contribute to the community network's affairs. Additionally, there are local groups, or social networks, formed by node owners within a specific zone or neighbouring zones. Members of local groups participate actively in zone related decisions, attending weekly meetings and collaborating through the respected local mailing lists. The organization and activity of such local groups is not strictly defined and is mainly driven by its members' interests, their available time and knowledge. In general, the main networking aspects such as address range allocation matters are resolved in the global community level since they affect the overall community network whereas local organisational aspects are discussed in the local group level and always driven by individuals.

2.3 Users

Members in community networks principally act as both consumers and producers. First, members can be considered as producers since they provide infrastructure and time to the networks. Furthermore, members can also be regarded as consumers since they utilize the network's available services. However, a community network is not solely based on infrastructure resource contributions. Users are also expected to contribute time and knowledge to manage those resources. Time is needed for instance for maintenance tasks (tasks which might require technical knowledge or not). Additionally, technical knowledge is needed as well to properly configure given IP network components. Community network contributions, both infrastructural and time or knowledge spent by users are purely voluntary, thus, no money rewards are offered. Typically, contributing members dedicate their limited free time at will to the community network. It should be noted that in Guifi.net member expertise varies significantly

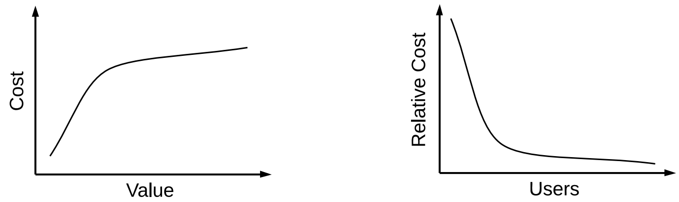

Figure 2.1: Relationship between cost and value in evolution of community cloud

throughout the owners of more than 20,000 nodes and a limited fraction has a technical or engineering background. Therefore, technical contributions originating from the community network itself can only be moderate. In general usage, members target off-the-self deployments of tools and applications to cover their needs.

2.4 Socio-Economic Context for Community Cloud System

We explore here the macroeconomic mechanisms [1] that can help in adoption and growth of community cloud model [2, 3]. The proposed architecture, see Figure 4.1, needs to take advantage of socio-economic context of community networks to ensure the success of community cloud model. We discuss first a cost-value proposition describing the conditions under which community clouds should emerge. Secondly, we propose a set of macroeconomic policies that, if placed in community networks, should accelerate the uptake and help the sustainability of community clouds.

2.4.1 Cost and Value Relationships in Community Cloud

Figure 2.1 shows the desired relationship between the cost and value proposition as the community cloud evolves and gets adopted by wider audience. In the nascent stage, the community cloud will not be able to provide much value until a critical mass of users are using the system. After that threshold, still the relative cost to achieve a little utility will be significant, which means that the early adopters of the system remain highly motivated and committed to the success of community cloud and continue to contribute resources even though they receive little value from the system in return. But once a significant proportion of the overall population has joined the community cloud, the relative cost to obtain value from the system tumbles and in the longer run the system is able to sustain itself with contributions that may be small in size but are made by a large number of users. The objective of the economic mechanisms and the social and psychological incentives is to let the system transition from inception through early adoption to finally ubiquitous usage.

2.4.1.1 Costs for Participation

The initial costs for setting up nodes in the community cloud involves hardware costs including the price of the computing and networking equipment, and installation costs including the manual labour needed. The continuous operation of the cloud node requires additional costs including network costs given by donating network bandwidth and any other subscription fees, energy costs to pay for electricity bills to run the computer equipment as well as cooling apparatus, maintenance cost to fund any technical support and replacements for parts, and hosting costs to provide storage space for the equipment. Besides these costs at the individual level, there are also the transaction costs [4] or management overheads to direct the group coordination and collaborative production efforts necessary for the operation of community cloud.

2.4.1.2 Value Proposition

The individuals in community cloud act as private enterprises where they offer services to generate revenue. The revenue for the community cloud users include tangible benefits like the services and applications that they will be able to consume, and intangible benefits like the sense of belonging to the community and personal satisfaction because of their contributions. The services can range from infrastructure to platform to software services meeting a spectrum of different needs of the users. Once community cloud gets adopted by a critical mass, community may also generate revenue by offering computing resources to commercial enterprises, similar to selling excess power capacity in the case of Smart Grid. For example, community can get into partnership agreements with the ICT providers where community can buy network bandwidth in return for providing access to the computing resources of the community cloud.

2.4.1.3 Comparison with Commercial Services

We discuss the community cloud cost and value in comparison with two popular commercial services that are also based in part on the idea of reciprocal sharing, Spotify[1] and Skype[2]. Spotify is a subscription-based music streaming service which reduces its infrastructure and bandwidth costs by serving cached content from users' devices as well as its own servers. Skype is a communication service which uses caches on users' devices for storing and processing information required for managing the underlying infrastructure. Both Spotify and Skype offer free as well as paid services. Why do users agree to contribute resources, and even when they are paying for the service?

An argument is that the costs for users are minimal. Both services mostly consume storage, computation time, power and bandwidth on the users' devices. Since these resources are not very expensive and the services' usage remains relatively low, the users do not mind this arrangement or not even notice it. But even more important, these services are designed so intuitively that most users do not even realise about donating the resources, and even when they do, the value these services provide has sufficient incentive.

The success of such services implies that for community cloud as well, the users should be able to join with zero or very little costs. The value proposition of the community cloud services should be strong enough to attract early adopters and keep them committed. The economic mechanisms in place for encouraging reciprocal sharing and ensuring overall system health and stability should be either invisible for non-technical users or very simple to understand and work with.

2.4.2 Macroeconomic Mechanisms for Architecture

We discuss in this section the macroeconomic policies we propose for community clouds, addressing relevant issues of the technical, social, economic and legal aspects of the community cloud system. We approach the problem by having explored some of the mechanisms previously in simulations [5] and also by developing a prototype implementation which is currently deployed in the Guifi community network [6] and which will allow to get users involved and participating in a real world scenario.

[1] http://www.spotify.com
[2] http://www.skype.com

2.4.2.1 Commons License

The agreement and license to join a community cloud should encourage and help enforce reciprocal sharing for community clouds to work. The Wireless Commons License[3] or Pico Peering Agreement[4] is adopted by many community networks to regulate network sharing. This agreement could serve as a good base for drafting an extension that lays out the rules for community clouds.

2.4.2.2 Peering Agreements

When different community clouds federate together, agreements should ensure fairness for all the parties. Agreements between different communities should describe the rules for peering between clouds. Within such agreements, local currency exchanges could be extended to address cases of imbalance in contribution across different zones [7].

2.4.2.3 Ease of Use

The easier it is for users to join, participate and manage their resources in the community cloud, the more the community cloud model will be adopted. This requires lowering the startup costs and entry barriers for participation. To this end, in terms of an institutional policy, we have developed a Linux-based distribution[5], to be used in the Guifi.net community cloud [6]. It will make the process of joining and consuming cloud services almost automated with little user intervention. This effect will make the community cloud appealing to non-technical users.

2.4.2.4 Social Capital

Community clouds need to appeal to the social instincts of the community instead of solely providing economic rewards. This requires maximising both bonding social capital [8] within local community clouds in order to increase the amount of resources and commitment of the users, and bridging social capital in order to ensure strong cooperation between partners in federated community clouds. Research on social cloud computing [9] has already shown how to take advantage of the trust relationships between members of social networks to motivate contribution towards a cloud storage service.

2.4.2.5 Transaction Costs

The community cloud, especially in its initial stages, will require strong coordination and collaboration between early adopters as well as developers of cloud applications and services, so we need to lower the transaction costs for group coordination [4]. This can take advantage of existing Guifi.net's mailing list[6], but also of the regular social meetings and other social and software collaboration tools. It also requires finding the right balance between a strong central authority and decentralised and autonomous mode of participating for community members and software developers.

[3] http://guifi.net/es/ProcomunXOLN
[4] http://www.picopeer.net
[5] http://repo.clommunity-project.eu
[6] http://guifi.net/en/forum

2.4.2.6 Locality

Since the performance and quality of cloud application in community networks can depend a lot on the locality, applications need to be network and location aware, but this also requires that providers of resources should honour their commitment to local community cloud implying that most requests are fulfilled within the local zone instead of being forwarded to other zones. We have explored the implications of this earlier when studying the relationship between federating community clouds [5, 10].

2.4.2.7 Overlay Topology

Community networks are an example of scale-free small-world networks [11], and the community cloud that results from joining community networks users is expected to follow the same topology and inherit characteristics similar to scale-free networks. As the overlay between nodes in the community cloud gets created dynamically [12], the community cloud may evolve along different directions as users of the underlying community network join the system. As the applications in community cloud will most likely be location and network aware to make the most efficient use of the limited and variable resources in the network, the overlay steered concentration and distribution of consumers and providers of services direct the state and health of the community cloud.

2.4.2.8 Entry Barriers

In order to control the growth of the community cloud and provide a reasonable quality of experience for early adopters and permanent users, different approaches can be considered, for example, a community cloud open to everyone, by invitation only, or one that requires a minimum prior contribution.

2.4.2.9 Role of Developers

The developers of the cloud applications are expected to play an important intermediary role between providers of resources and consumers of services, for example adding value to the raw resources and selling them to consumers at a premium. End users could have both the roles of raw resource providers and consumers which find the value of the cloud in the provided applications.

2.4.2.10 Service Models

Cloud computing offers different service levels, infrastructure, platform and software-as-a-service (SaaS). Similar to the three economic sectors for provisioning goods, the third level, the SaaS of the cloud reaches the end users. For providing value from the beginning in the community cloud, we propose to prioritize provisioning SaaS at the early stage of the community cloud.

2.4.2.11 Value Addition and Differentiation

The community cloud requires services that provide value for users. In addition, these services need to compete and differentiate from the generic cloud services available over the Internet. In this line, FreedomBox[7] services focus on ensuring privacy, and FI-WARE CoudEdge[8] and ownCloud[9] let cloud

[7] http://freedomboxfoundation.org
[8] http://catalogue.fi-ware.eu/enablers/cloud-edge
[9] http://owncloud.org

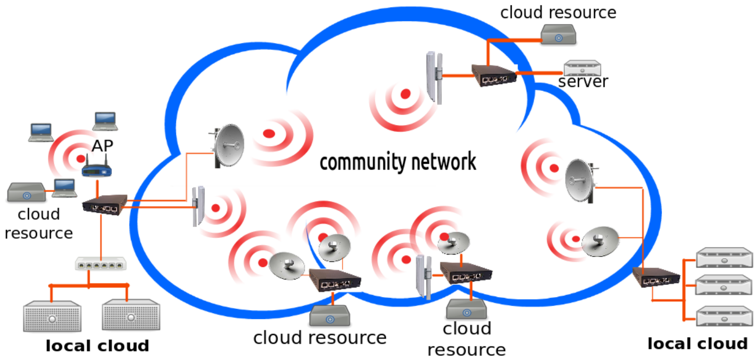

Figure 2.2: Nodes in a community network

applications consume resources locally.

2.5 Nodes and topological aspects of community networks

A community network is managed and owned by the community, where nodes are managed independently by their owners. The computer machines or nodes in a community network vary widely in their capacity, function and capability, as illustrated in Figure 2.2. Some hardware is used as super nodes (SN) that have multiple wireless links and connect with other SNs to form the backbone of the community network, and are usually intended to be stable with permanent connectivity. Most SNs are installed in the community network participant's premises. A few SNs, however are placed strategically in a third party location, e.g. telecommunication installations of municipalities, to improve the community network's backbone. Other nodes in the community network act as ordinary nodes (ON) and are only connected to the access point of a SN. Topological analysis of the Guifi.net community network [11] indicates that from approximately 17,000 analysed nodes of Guifi.net, 7% are SNs while the others are ONs.

From the node types shown in Figure 2.2, it can be seen that principally the hardware for computation and storage is already available in community networks, consisting of some servers attached to the networking nodes. No cloud services, however, are yet deployed in community networks to use this hardware as a cloud, leaving the community network services significantly behind the current standard of the Internet. Our vision is that some community wireless routers will have cloud resources attached, building the infrastructure for a community cloud formed by several cloud resources attached to the nodes. We note that ONs could principally also contribute cloud resources.

Figure 2.3 shows the outdoor view of a community network SN. The equipment, mainly antennas and radios, is used for building wireless links between other SNs. Figure 2.4 shows an example of the indoor hardware of a SN. The router used is a Microtik RB750, while a Jetway JBC362F36W with Intel Atom N2600 CPU, 2GB RAM and 64GB USB has been added to become a cloud resource for the community cloud. A laptop is used as an additional server, while a UPS keeps the node running in the case of power failure.

Figure 2.3: SN with outdoor hardware for wireless links

Figure 2.4: Indoor hardware of a community network node with router, server and cloud resource

3 Requirements for Community Cloud System

A community cloud is a combination of a number of cloud systems being run and managed independently by the different community members. The community cloud bridges in different aspects the gap between the public cloud, the general purpose cloud available to everyone, and the private cloud, available to only a limited set of users with user-specific services. These requirements provide the foundation for the design of the community cloud system, and need to be satisfied for it to be deployed and adopted successfully by the community.

3.1 Autonomy

Community cloud systems may be formed based on individual cloud systems that are set up and managed independently by different owners. The main requirement for a cloud owner for participating in such a community cloud is that the local cloud setup should adhere to the common API provided by the community cloud, and contribute resources to the community.

3.2 Security

There are many security challenges that need to be addressed for ensuring users' trust in the system, and with multiple independent cloud providers from the community, security becomes even more important in a community cloud. For instance, the data and applications running on different cloud systems should be protected from unauthorised access.

3.3 Self-Management

Community cloud should self-manage itself and continue providing services without disruption when nodes go offline. Self-management should also help in the coordination between different cloud owners that become part of a federated community cloud.

3.4 Utility

For the acceptance of the community cloud, it should provide applications that are valuable for the community, since usage strengthens the value of the community cloud, motivating its maintenance and update. These applications need to differentiate from the generic cloud services available over the Internet.

3.5 Ease of Use

Most of the users of the community cloud will not be proficient in cloud technologies, so setting up nodes for deployment and managing cloud software should be simple and straightforward. The easier it is for users to join, participate and manage their resources in the community cloud, the more the community cloud model will be adopted. To this end, in terms of an institutional policy, we have developed a Linux-based distribution for deployment in the Guifi.net community cloud [6]. It will make the process of joining and consuming cloud services almost automated with little user intervention.

3.6 Incentives for Contribution

Community cloud builds upon collective efforts of the members of the community networks, and requires the contribution of the volunteers in terms of their time, knowledge and effort as well as computing, storage and network resources. For the community clouds to be sustainable, incentive mechanisms are needed to encourage users to actively contribute towards the system.

3.7 Support for Heterogeneity

The hardware and software used by members in a community cloud can have quite varying characteristics, and the cloud system should handle this seamlessly.

3.8 Standard API

The cloud system should make it straightforward for the application programmers to design their applications in a transparent manner for the underlying heterogeneous cloud infrastructure. The API should provide the appearance of a middleware that obviates the need to customize the applications specific to each cloud architecture. This is essential for community clouds when these result from the federation of many independently managed clouds. Providing a standard API for the community cloud ensures that applications written once for a particular community cloud system can be easily deployed on new cloud architectures.

3.9 QoS and SLA Guarantees

The community cloud system needs mechanisms for ensuring quality of service (QoS) and enforcing service level agreements (SLA).

4 Architecture

We foresee realising the community cloud by deploying a community cloud platform tailored to the specific infrastructure and context of community networks. A standard cloud platform is usually a centralized platform designed to perform resource management. There are quite a few well known cloud platforms for managing public and private clouds, like OpenStack[1], and OpenNebula[2], among others. In our effort nevertheless, we focus on providing a framework that would allow users to share resources and access collaboratively-built services. For instance, a community cloud platform would require incentive mechanisms inspired by the social nature of community networks integrated into resource regulation components to encourage contribution from the members of the community network [13]. In this section, we describe the architecture of our proposed framework that enables Cloud services on top of a Community network. First, we explain the layered conceptual architecture and following that, in the next chapter, we describe Clommunity distribution Cloudy as the core of the framework.

4.1 Layers of Community Cloud System

We propose a framework that can serve as the core of a community cloud system. Our community cloud framework is a distributed bottom-up resource sharing and collaborative services platform. This is achieved by adopting a layered architecture, as shown in Figure 4.1.

4.1.1 Hardware layer

Hardware layer provides the physical infrastructure needed to run the cloud services and applications. The hardware layer mostly consists of SNs, client nodes, routers and the communication infrastructure, along with any computation, storage and other resources attached to the nodes. It can be either bare metal, e.g., Clommunity boxes or servers that run OpenStack. In the latter, VMs are provisioned to provide infrastructure for Clommunity nodes. Clommunity resources are connected through the same community network. Resources from other networks can also become a part of the community by using VPN.

4.1.2 Core layer

In the core layer comes CLOMMUNITY distribution Cloudy, which is explained in section 5. This layer is responsible for managing the hardware as virtualised resources. It consists of components like manager for the hosts and the network, and controller, scheduler, monitor, and data store for virtual instances. Many popular open-source software can be integrated to provide virtualisation, for instance LXC (Linux Containers), OpenVZ, Docker and etc.

[1] http://www.openstack.org
[2] http://www.opennebula.org

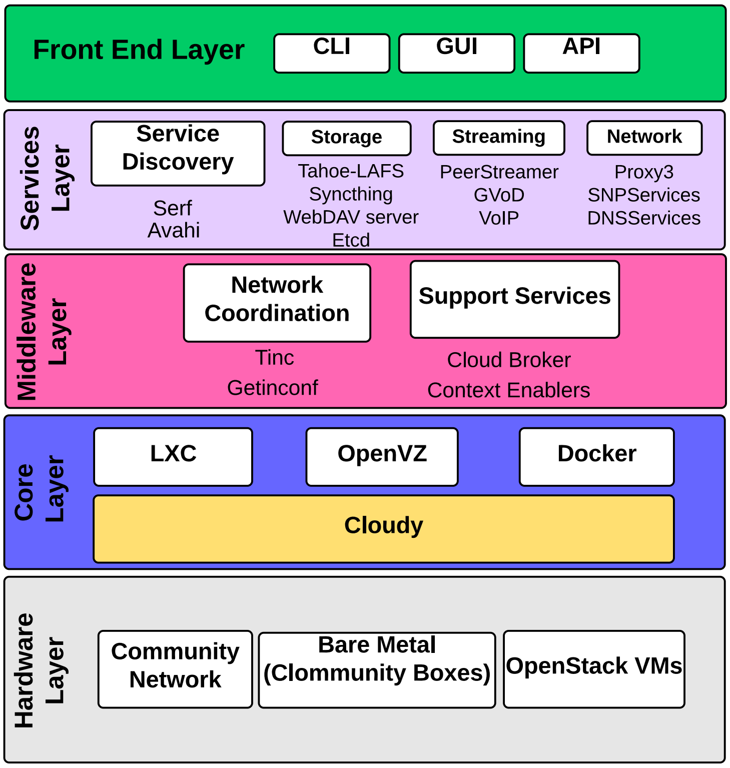

Figure 4.1: Community cloud framework

4.1.3 Middleware layer

In addition to virtualization, Cloudy enables to discover other Cloudy nodes. This is done by providing a decentralized virtual Layer 2 networking service over community network's Layer 3. Middleware layer also federates the resources from multiple local community clouds, providing an integrated and consistent view of the cloud system to the cloud services. This requires network coordination component to identify and manage different local clouds. Other support services can include cloud coordinator and services broker components for assisting combining resources from multiple cloud providers, and social and economic context enablers [14] that take advantage of the social and economic characteristics of the community networks to encourage participation from community network members and ensure sustainability of the community cloud model.

4.1.4 Services layer

Services layer integrates useful services and applications providing utility for the community network members to encourage their participation. In this layer, a customized version of both Serf and Avahi are available to enable decentralized service discovery. Common services include storage, video streaming, video on demand, voice over IP and network applications.

4.1.5 Front-end layer

Front-end layer provides the interface to interact with the infrastructure of the community cloud, including command line interfaces (CLI), graphical user interfaces (GUI), application programming interfaces (API), and any other tools for assisting development of cloud services and applications. It includes a service browser web interface that enables user to view all the available services in one place.

4.2 Community Cloud Network

Deployment of Cloud services on top of a community network can introduce a significant traffic growth to the network. This can decrease the network performance and consequently affect the users experience. We have foreseen such a problem and have proposed solutions to improve the Cloud network performance. Our solutions recommend network aware services and proximity to the users. In the rest of this section, we summarize our research results for improving the Cloud network performance.

4.2.1 Local and Federated Micro Clouds

We take the socio-technical context of community network into accounts and consider scenarios of local and federated Community Cloud. For the available services locally, communications between the nodes in the local community Cloud are sufficient. However, in the absence of a service or lack of resources in the nodes of a local Cloud, inter-Cloud communications are necessary.

4.2.1.1 Local Micro Community Cloud

This scenario is derived from the topology of the community network, given by the fact that the community network generally has two different types of nodes, SNs and ONs, and the observed characteristics of the strength of social network within zones [11]. In such a local community cloud, a SN is responsible for the management of a set of attached nodes contributing cloud resources. From the perspective of the attached nodes, this SN acts as a centralized unit to manage the cloud services.

4.2.1.2 Federated Community Cloud

Multiple SNs from different zones in a community network can connect and form federated community cloud [15]. SNs connect physically with other SNs through wireless links and logically in an overlay network to other SNs that manage local clouds. SNs coordinate among themselves for provisioning infrastructure service so the requests originating from one SN's zone can be satisfied by the resources allocated from another SN's zone. Figure 4.2 shows an example of a federated community cloud formed by SNs from three zones. The ONs in a given zone are directly managed by the SN in that zone but they can also consume resources from other zones because of the coordination among SNs.

4.2.2 Partitioning of Geo-Distributed Resources

We provide a solution for partitioning a distributed infrastructure into a set of computing clusters. Each cluster is called a *micro data center*. Micro data centers are a set of geo-distributed resources, each having a Cloud manager and able to host Cloud applications. A *Micro Data Center Recommender* component can be added to Cloudy in the Middleware layer as a support service. Every node will join to the micro data center recommended by the recommender service.

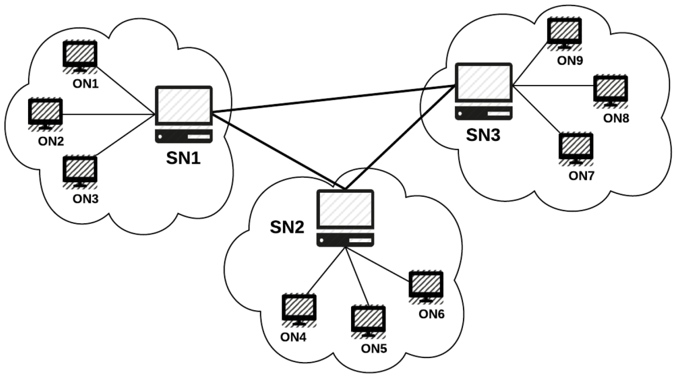

Figure 4.2: Nodes in federated community cloud

To form multiple micro data centers, we propose a novel community detection algorithm, which does not require to know the complete topology of the network. Our community detection algorithm is based on multiple biased random walks and discovers clusters of nodes that have high connectivity. Consequently, we improve the quality of Cloud services and increase the the network performance (e.g., bandwidth and latency). We explain our work in detail in D3.3[3], Section 2.1 auto-configuration of multiple Cloud data centers in Clommunity.

[3]D3.3 Experimental Research on Community Clouds

5 Cloudy: Implementation of Community Cloud System

Cloudy [16] is a distribution based on Debian GNU/Linux, aimed at end users, to foster the transition and adoption of the community network cloud environment. Cloudy is the implemented prototype of our community cloud framework described in Section 4.1. The current prototype of Cloudy implements the modules/layers shown in Fig. 4.1. A Cloudy instance can be run directly on a bare metal machine or on a virtual machine. Independently of the hardware that Cloudy runs on, connectivity to other Cloudy instances is obligatory in order to fully exploit the potential of Cloudy.

Cloudy consists of a number of services, designed to help building cloud-based services in community networks. Cloudy's main components can be considered as a layered stack with services residing both inside the kernel and higher up in the user-level. All of the software included in the Cloudy platform is open-source. All services can be accessed and managed through a main panel of Cloudy GUI. The following three groups classify the Cloudy services.

5.1 Infrastructure services

Virtualization is the main enabling technology for cloud computing. As such, providing community network users the resources to deploy virtual machines with a few clicks is a very convenient way to bring the cloud closer to their premises. This allows the non-experienced user to focus on the services and applications themselves rather than on learning how to cope with the underlying infrastructure.

OpenVZ [17] is an operating system-level virtualization technology for Linux based on containers. OpenVZ allows creating multiple secure, isolated operating system instances called containers (commonly known as VPSs) on a single physical machine enabling better server utilisation and ensuring that applications do not conflict with each other. Each container performs and executes exactly like a stand-alone server (it can have root access, users, IP addressing, memory, files, etc.) and can be started and stopped independently from the others and from the host machine. OpenVZ is the preferred solution for providing virtual machines in Cloudy with low-mid-end hardware as only a negligible part (1-2%) of the CPU resources is spent on virtualization. The Cloudy distribution includes a script that downloads and installs all the required OpenVZ packages in one click and through OpenVZ Web Panel Cloudy instances can be run in the VMs created using OpenVZ Web Panel.

Other virtualization methods used in Cloudy are *Docker* [18] and *LXC* [19]. This approach adds a special support for infrastructure services (IaaS), as the cloud nodes are able to create multiple virtual machine instances for other purposes in addition to the ones dedicated to Cloudy. The infrastructure services of Cloudy enable resource-sharing inside the CN.

5.2 Service discovery and network coordination services

As a Linux distribution, Cloudy lies at the lowest layer of the stack, providing custom decentralized network services for service discovery and service announcement. Service discovery and announcement are crucial building blocks for enabling distributed services to be orchestrated to provide platform and application services.

Cloudy includes a customized version of *Avahi* [20] a service discovery tool, to provide decentralized service discovery at layer 2, needed to discover other services that will be used to provide higher-level services. However, the multicast based design does not allow the Avahi service to reach beyond the local link, which is the case in community networks, where services are spread over different nodes that belong to different broadcast domains. In this environment, Avahi packets can not be directly exchanged from one node to another. For this reason, we are using *TincVPN* [21] a Virtual Private Network (VPN) daemon that uses tunneling and encryption to create a secure private layer 2 network between hosts of different domains. This layer 2 connectivity is needed between nodes, since they may reside on different administrative domains and even be located behind firewalls. TincVPN is automatically installed and configured on every Cloudy node, ready to be activated (for privacy reasons, this option is left to the users' choice). After its activation, a VPN daemon is started in order to reach other Cloudy instances via the established layer 2 network and thus Avahi can communicate transparently with other nodes. To easily install and configure a system with TincVPN, a tool called *Getinconf* [22] has been developed, which is integrated in Cloudy.

Cloudy includes also *Serf* [23] a lightweight tool to announce and discover services in the community networks. Serf is a decentralized solution for cluster membership, failure detection, and orchestration. It relies on an efficient and lightweight gossip protocol to communicate with other nodes that periodically exchange messages between each other. This protocol is, in practice, a fast and efficient way to share small pieces of information. An additional byproduct of having this service distributed all over the community network cloud is that it allows to evaluate the quality of the point-to-point connections between different Cloudy instances. This way, Cloudy users can decide which service provider to choose based on network metrics like RTT (round-trip-time), number of hops or packet loss. The combination of Avahi, TincVPN, Getinconf and Serf in Cloudy implements the coordination of the resources and the services in the community network cloud.

5.3 User services

5.3.1 Platform as a Service (PaaS)

Providing attractive platform services to community members as distributed filesystem, highly available key-value store, file synchronization, video streaming, video-on-demand, VoIP, NAT traversal support and many more is of high importance.

On of the promising services for storage is *Tahoe-LAFS* [24]. Tahoe-LAFS is a free open and secure cloud storage system. It distributes the data across multiple servers. It allows community users to share their storage with other members. A 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 get the list of all connected storage nodes [25]. The configuration of Tahoe-LAFS and the process of deploying a whole storage grid is assisted by the Avahi service discovery tool using the web interface of Cloudy, where the user only needs to introduce some basic information.

Etcd [26] a highly-available key value store for shared configuration and service discovery and *Syncthing* [27] an open source file synchronization client/server application are already included in the Cloudy distribution.

5.3.2 Software as a Service (SaaS)

Cloudy allows the user services to be present inside the community network and to be easily deployed and managed via the Cloudy interface. Users can deploy their preferred services and share them with others. One of such multimedia services included in Cloudy is *PeerStreamer* [28], an open source live streaming platform. PeerStreamer includes a streaming engine for the efficient distribution of media streams, a source application for the creation of channels and a player applications to visualize the streams. Streaming is assisted by Cloudy, by supporting the user to publish a video stream or connect to a peer (assisted by Serf or Avahi). Services that enables users to find and watch video content on-demand at any time like *Gvod* [29], a decentralized search service as *Sweep* [30] and a distributed key-value store as *CaracalDB* [31] are additional services that are part of the Cloudy.

6 Conclusions and Outlook

In order to enable a bottom-up Community Cloud system built by the community users, we have proposed a layered network and services architecture. The bottom layer is the hardware layer, which are Community Cloud resources connected through community network. These resources are provided by the users. We implemented a Community Cloud distribution, named Cloudy, that provides utility and services required to build the Cloud services. Cloudy's main components can be considered as a layered stack with services residing both inside the kernel and higher up in the user-level. All of the software included in the Cloudy platform is open-source. All services can be accessed and managed through a main panel of Cloudy GUI. The following three groups classify the Cloudy services.

The deliverable D3.4 is the final revised version of D3.1 delivered in M06. D3.1 contains the requirements for the network and service architecture, and presented already a first version of the architecture itself. Building upon D3.1, we discuss and extend the final network and service architecture, including a layered architecture and its implementation, based on the insights we have obtained from results of WP4 and WP2.

Bibliography

[1] Amin M Khan and Felix Freitag, "Exploring the Role of Macroeconomic Mechanisms in Voluntary Resource Provisioning in Community Network Clouds," in *Distributed Computing and Artificial Intelligence, 11th International Conference*, vol. 290 of *Advances in Intelligent Systems and Computing*, pp. 269–278. Springer International Publishing, Salamanca, Spain, June 2014. 2.4

[2] Peter Mell and Timothy Grance, "The NIST Definition of Cloud Computing," *NIST Special Publication*, vol. 800, no. 145, 2011. 2.4

[3] Alexandros Marinos and Gerard Briscoe, "Community Cloud Computing," in *Cloud Computing, First International Conference, CloudCom 2009*, MartinGilje Jaatun, Gansen Zhao, and Chunming Rong, Eds., vol. 5931 of *Lecture Notes in Computer Science*, pp. 472–484. Springer Berlin Heidelberg, Beijing, China, Dec. 2009. 2.4

[4] R. H. Coase, "The Nature of the Firm," *Economica*, vol. 4, no. 16, pp. 386–405, Nov. 1937. 2.4.1.1, 2.4.2.5

[5] Amin M Khan, Umit Cavus Buyuksahin, and Felix Freitag, "Towards Incentive-based Resource Assignment and Regulation in Clouds for Community Networks," in *Economics of Grids, Clouds, Systems, and Services*, Jörn Altmann Altmann, Kurt Vanmechelen, and Omer F. Rana, Eds., vol. 8193 of *Lecture Notes in Computer Science*, pp. 197–211. Springer International Publishing, Zaragoza, Spain, Sept. 2013. 2.4.2, 2.4.2.6

[6] Javi Jiménez, Roger Baig, Felix Freitag, Leandro Navarro, and Pau 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, Mar. 2014, IEEE. 2.4.2, 2.4.2.3, 3.5

[7] Magdalena Punceva, Ivan Rodero, Manish Parashar, Omer F. Rana, and Ioan Petri, "Incentivising resource sharing in social clouds," *Concurrency and Computation: Practice and Experience*, Mar. 2013. 2.4.2.2

[8] James S Coleman, "Social capital in the creation of human capital," *American Journal of Sociology*, vol. 94, no. 1988, pp. S95—-S120, 1988. 2.4.2.4

[9] Kyle Chard, Kris Bubendorfer, Simon Caton, and Omer F. Rana, "Social Cloud Computing: A Vision for Socially Motivated Resource Sharing," *IEEE Transactions on Services Computing*, vol. 5, no. 4, pp. 551–563, Jan. 2012. 2.4.2.4

[10] Amin M Khan, Umit Cavus Buyuksahin, and Felix Freitag, "Prototyping Incentive-based Resource Assignment for Clouds in Community Networks," in *28th IEEE International Conference on Advanced Information Networking and Applications (AINA'14)*, Victoria, Canada, May 2014, IEEE. 2.4.2.6

[11] Davide Vega, Llorenc Cerda-Alabern, Leandro Navarro, and Roc 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. 612–619, IEEE. 2.4.2.7, 2.5, 4.2.1.1

[12] Akihiro Nakao and Yufeng Wang, "On Cooperative and Efficient Overlay Network Evolution

Based on a Group Selection Pattern," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 40, no. 2, pp. 493–504, Apr. 2010. 2.4.2.7

[13] Amin M. Khan, Umit Cavus Buyuksahin, and Felix Freitag, "Incentive-based Resource Assignment and Regulation for Collaborative Cloud Services in Community Networks," *Journal of Computer and System Sciences*, , no. 2014, Dec. 2014. 4

[14] Amin M Khan, Mennan Selimi, and Felix Freitag, "Towards Distributed Architecture for Collaborative Cloud Services in Community Networks," in *6th International Conference on Intelligent Networking and Collaborative Systems (INCoS'14)*, Salerno, Italy, Sept. 2014, IEEE. 4.1.3

[15] Rafael Moreno-Vozmediano, Rubén S. Montero, and Ignacio M. Llorente, "IaaS Cloud Architecture: From Virtualized Datacenters to Federated Cloud Infrastructures," *Computer*, vol. 45, no. 12, pp. 65–72, Dec. 2012. 4.2.1.2

[16] "Cloudy: A community networking cloud in a box," http://cloudy.community/, 2015. 5

[17] "OpenVZ Linux Containers," http://openvz.org/, 2015. 5.1

[18] "Docker Containers," https://www.docker.com/, 2015. 5.1

[19] "LXC: Linux Containers," https://linuxcontainers.org/, 2015. 5.1

[20] "Avahi Service Discovery Tool," http://avahi.org/, 2015. 5.2

[21] "TincVPN," http://tinc-vpn.org/, 2015. 5.2

[22] "Getinconf tool," https://github.com/Clommunity/getinconf. 5.2

[23] "Serf," https://www.serfdom.io/, 2015. 5.2

[24] Zooko Wilcox-O'Hearn and Brian Warner, "Tahoe: The least-authority filesystem," in *Proceedings of the 4th ACM International Workshop on Storage Security and Survivability*, New York, NY, USA, 2008, StorageSS '08, pp. 21–26, ACM. 5.3.1

[25] Mennan Selimi and Felix Freitag, "Tahoe-LAFS Distributed Storage Service in Community Network Clouds," in *4th IEEE International Conference on Big Data and Cloud Computing (BDCloud 2014)*, Sydney, Australia, Dec. 2014, IEEE. 5.3.1

[26] "Etcd key-value store," https://github.com/coreos/etcd, 2015. 5.3.1

[27] "Syncthing," http://syncthing.net/, 2015. 5.3.1

[28] "PeerStreamer: P2P Media Streaming," http://peerstreamer.org/, 2015. 5.3.2

[29] Gautier Berthou and Jim Dowling, "P2p vod using the self-organizing gradient overlay network," in *Proceedings of the Second International Workshop on Self-organizing Architectures*, New York, NY, USA, 2010, SOAR '10, pp. 29–34, ACM. 5.3.2

[30] "Sweep," http://wiki.clommunity-project.eu/pilots:sweep, 2015. 5.3.2

[31] "CaracalDB," https://github.com/CaracalDB/CaracalDB, 2015. 5.3.2

Licence