A Storage Service based on P2P Cloud System

Gerardo García-Rodríguez, Francisco de Asís López-Fuentes

Departament of Information Technology
Universidad Autónoma Metropolitana-Cuajimalpa (UAM-C)
Av. Vasco de Quiroga 4871, Cuajimalpa, Santa Fe
05348 Mexico City, México
{flopez}@correo.cua.uam.mx

Abstract. During the last years, cloud computing has been quickly adopted worldwide, and several solutions have emerged. Cloud computing is used to providing storage service, computing power and flexibility to end-users, in order to access data from anywhere at any time. Although, all benefits introduced by cloud computing, this technology still faces several challenges related to privacy, security, robustness, scalability and throughput. Many of these problems are due to the fact that traditional cloud computing is based on centralized servers, who are responsible for the management and storage all data. This paper proposes a P2P infrastructure as alternative platform for deploying cloud storage services. In particular, we are interested in exploring cloud computing for personal storage services.

Keywords: P2P architectures, cloud computing, distributed systems.

1 Introduction

Cloud computing has become more and more popular every day, and many users and companies use these services to store their data or to get more computing power. According to M. Armbrust et al [1], cloud computing refers to the applications delivery and services over the Internet, as well as the hardware and systems software in the datacenters that provide these services. From this definition we can deduce that cloud computing is a model that allows access to files, applications or services in a ubiquitous and pervasive way through network in order to share a set of configurable computing resources. These resources can be servers, storage, applications and services, which can be rapidly provisioned and released with a minimal effort in service management or interacting with the provider. Therefore, cloud computing provides the illusion of unlimited and on-demand scalability. Authors in [2] identify the following essential characteristics of a Cloud: on-demand self-service, network access, resource pooling, elasticity and measured service. Cloud computing refers to two very basic concepts: abstraction and virtualization [5]. Abstraction means that the implementation details of the system users and developers are abstracted. Thus, applications run on physical systems that are not specified, the files are stored in places where users do not know their actual location, the system can be managed via outsourcing, and clients can access to the system in a ubiquitous manner. On the other hand, resources of the systems are pooled and shared in a virtualized way. Regarding virtualization, Sosinsky, B. states in

[5] that systems and storage can be provisioned as needed from a centralized, costs are assessed on a metered basis, multi-tenancy is enabled, and resources are scalable with agility.

Although cloud computing introduce several benefits such as massive computing power, storage capacity and great flexibility, cloud computing is facing challenges. Most traditional cloud computing systems are centralized and based on the client-server paradigm. A centralized structure introduces several drawbacks in cloud computing as storage dependence, privacy, scalability, privacy locally or connectivity. In this scenario, peer-to-peer (P2P) networks have emerged as a promising distributed information management platform. A P2P network is a distributed network formed by a group of nodes, which build a network abstraction on top of the physical network, known as an overlay network. In a P2P system each peer can take the role of both, a server and of a client at the same time. An important advantage of P2P networks is that all available resources such as processing, memory and bandwidth are provided by the peers. Therefore, when a new peer arrives to the P2P system the demand is increased, but the overall capacity too. This is not possible in a client server model with a fixed number of servers. P2P paradigm allows that a distributed platform distributes its load and duties on the participating peers. In this paper, we propose a general architecture for cloud computing services based on P2P networks. Main characteristic of this project is its qualified storage method based on reliability indices, which are totally transparent to the user as it is in the centralized cloud computing.

The rest of this paper is organized as follows. In Section 2, we introduce the main concepts related to cloud computing and peer-to-peer networks. We introduce our proposed in Section 3. Then, in section 4, we explain the implementation of our model and its evaluation. This paper concludes in Section 5.

2 Background

A distributed storage system is an infrastructure that allows to store files in nodes, which are connected through a computer network. These systems are characterized by their wide range of applications, that among most important are backup files, sharing of files in network and edition of documents from different locations. On the other hand, P2P paradigm has also been used to build distributed storage systems. The basic idea of distributed file system by P2P is that we can replicate our personal files and save these replicas in the shared spaces of other network users, as well as sharing of resources such as hard disk, processing via a P2P network and users' interconnection. Users must accept the pieces of files on their part of the network shared disk, file redundancy to have multiple backups and user registration. In this type of system, user can know where the files are stored. In contrast, a P2P storage system only supports flat namespaces, distributed file systems typically support hierarchical namespaces [3]. Cloud computing typically has been used as a storage system. Many specialists separate the cloud computing into two different models: service and development. A service model defines the level of abstraction at which a customer interfaces a cloud computing environment [4], while a development model refers to the location and management of the cloud infrastructure. There are three types of service accepted and defined, which are the "Software as a Service" (SaaS) model, the "Platform as a Service" (PaaS) model,

and the “Infrastructure as a Service" (IaaS) model. SaaS is an entire operating environment with applications, management and user interfaces. In this model, the application is given by the client through a small user interface, which could be a web browser, and the clients has an absolute responsibility to manage their files only. The entire application is the responsibility of the supplier [5] and the Cloud customer has no control over this infrastructure. A PaaS Cloud provides virtual machines, operating systems, application services, development frameworks, transactions and control structures for applications developed by the Cloud customer. In this model, the users or customers can develop their applications within the cloud infrastructure or use their applications scheduled. The service provider manages the cloud infrastructure, operating system, software or enabling [5]. However, the customers are responsible for the installation and maintenance of the applications they are developing. Finally, an IaaS Cloud provides fundamental computing capabilities such as virtual machines, virtual storage, virtual infrastructure and other hardware analogy as a provision for customers. IaaS provider manages all infrastructure while the customer is responsible for the development aspects [4], [5]. Respect to the development model we have the following types [5]: private, public, community and hybrid. Recently has emerged the P2P cloud concept, which combines cloud and P2P networks. This type of Cloud computing is based on a fully distributed Cloud architecture and can be useful for some usage scenarios. Authors in [4] state that a P2P Cloud allows organizations or even individual to build a computing infrastructure out of existing resources, which can be easily allocated among different tasks. Potential benefits of P2P Cloud computing have recognized during the last year and several related work have been proposed. Cloud@Home is presented in [6], which is a hybrid system that combines characteristics from volunteer computer and cloud computing paradigms. Xu et al [8] propose another distributed P2P Cloud system which is designed to provide storage service only. An architecture and its prototype to provide an infrastructure and service through a P2P cloud are presented in [4]. Authors in [7] combine P2P and cloud computing to obtain a hybrid and distributed architecture for multimedia streaming service. This work is focused mainly to reach QoS requirements.

3 Proposed Model

This section introduces our model. We propose a distributed storage service that takes advantage of P2P networks in order to create a cloud based on this type of infrastructure, that is to say, stores files are decentralized and its location is completely transparent to the user. This proposal differs from traditional distributed file systems which are equally used to share resources, but they have a lower level of transparency, and users know about the file system. A P2P Cloud system consists of peers which contain the client application. Peers come together to share their available hard disk capacity in order to create a total storage space together. The contents are split and each of these fragments are distributed within the peers. If a content is not split, then it only is sent by the tracker to any of the nodes that is part of the cloud. Thus, a user can realize a replication of files within the cloud.

Reputation levels play an important role, because they allow the system to define the way how files are stored within the peers. When a tracker is created, it must contain the table of confidence (trust) levels which consists of an average of the levels of trust, dynamism and availability of each peer in the system. The tracker decides where to store the files according to the level of reputation of each peer. We exemplify this case as follows: if the peer P1 is level 5, then it can store its content on peers with same reputation level. In contrast, if P1 has a low reputation level its content is sent to peers with the same level of reputation. We use this strategy for the benefit of the cloud, and thus any type of contingency in the cloud as file access failure by peers with greater reliability. Figure 1 illustrates the tracker operation in the cloud. The tracker or server routes data and establishes communication between peers. These communications are based on the confidence level of each peer and contain the information required for each.

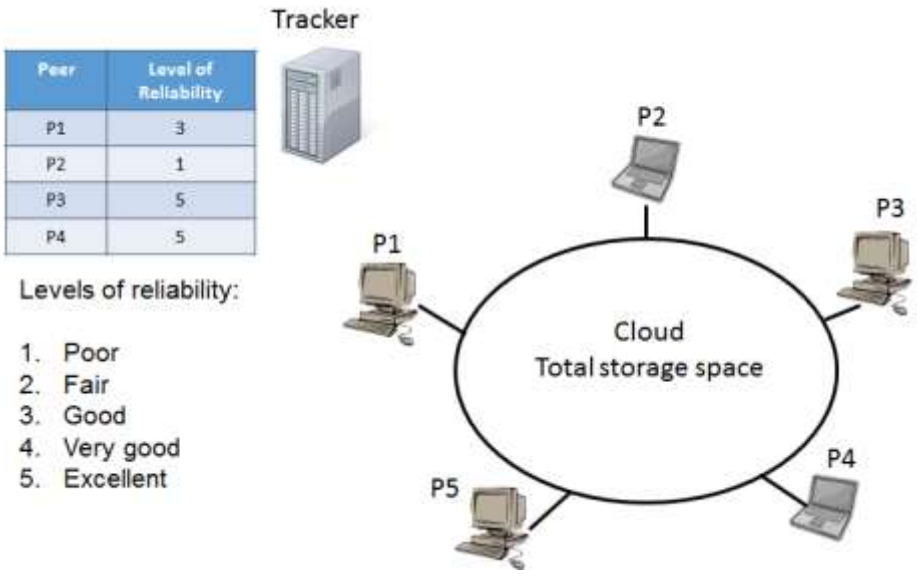

Fig. 1. Proposed P2P cloud model

Figure 2 shows how the peer P5 uploads a file to the cloud. We can see how the tracker is responsible for routing the file to a peer with the same level of reliability as P5. In this case tracker selects peers P3 to store the file of peer P5, because both peers have same level of reliability.

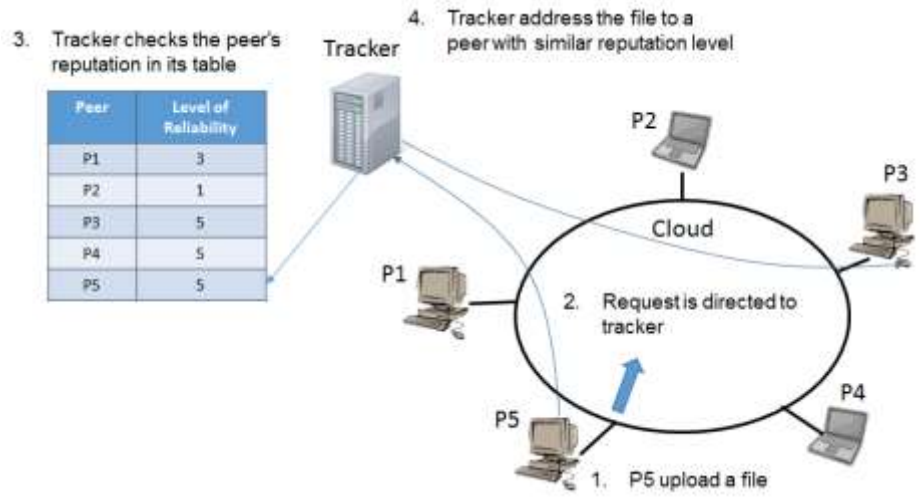

Fig. 2. Operation example for storing a file

4 Implementation

We implement our model using different entities and addressing methods. These entities are peers, tracker and database, which are involved in the cloud storage system and interaction between them.

Peer. This application is hosted on each node in the P2P network, and is executed in each computer involved in the P2P cloud. Since a peer is an entity that receive and send files at the same time, we need to implement an application to perform these two tasks simultaneously. These tasks are realized through the peer application. Peer application also is responsible for monitoring each peer and reports its shared resources to tracker. Parameters to be monitored in each peer are its store capacity (trust), the number of disconnections during a day (availability), and its availability (number of storage requests that are rejected by a peer). These three metrics are averaged in order to obtain the reliability level in each peer. After all, reliability is the metric used by a tracker to route content between peers. Peer application is formed by two parts: a server and a client. Server part (server.c) always is listening in order to attend to other peers. In this case, to store files in the host computer. Our server application is composed by the following functions *receive_file, exceed_client, send_file, reloj* and *create_directory*. A flow diagram for our peer application (server side) is depicted in Figure 3. On the other hand, client part realizes different functions such as uploading files, display files and exit. Additionally, client program is who communicates with tracker in order to report all information about this computer in the cloud system. Client application (client.c) is composed by the following functions: *send_direction, send_file, notify_output, tie_file_location, download_file.* Figure 4 shows how these functions work in the client application.

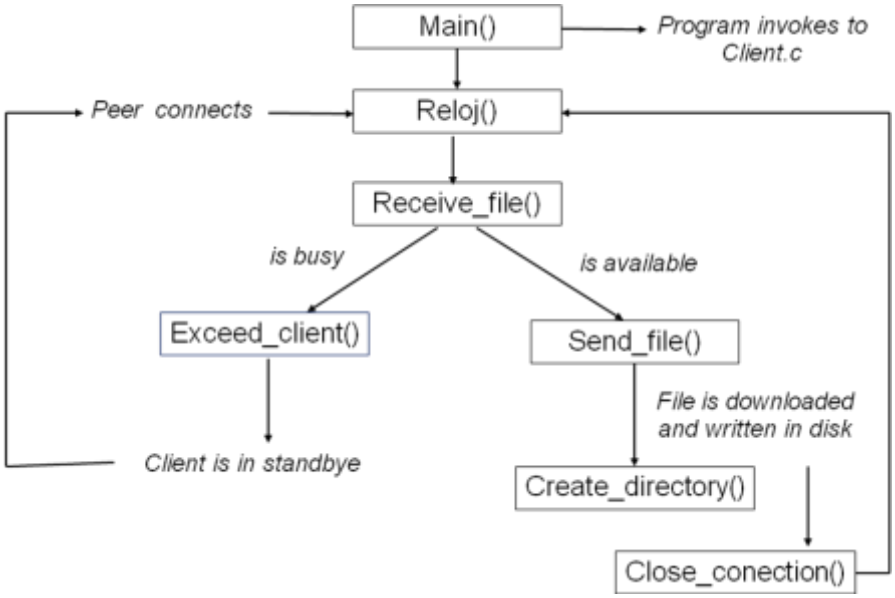

Fig. 3. Flow diagram for Peer application

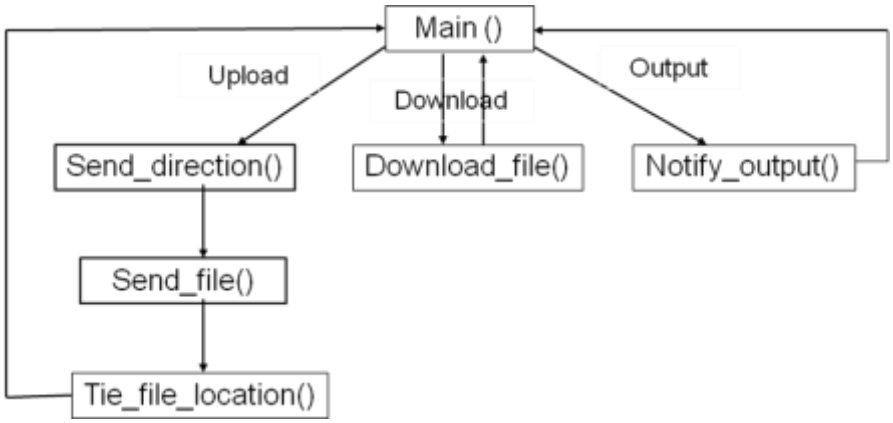

Fig. 4. Flow diagram for Client application

Tracker is responsible for system initialization. This application establishes the communications among peers. Tracker application also manipulates the database, in which the reports generated by each peer and its contents are registered. It is important to note that the tracker is not a storage server, so the tracker never manage files. Tracker

only redirects the contents received from a peer to another depending on levels of reliability of each peers. However, tracker offers localization transparency to all users in the systems. In this way, a peer sends its content to tracker, whom decides in which peer will be placed this content. This location is transparent for the requesting peer. When a requesting peer whishes recovery its content, it is requested via the tracker. Then, content is be addressed from the host peer to the requested peer. To allocate contents, the tracker queries the levels of dynamism, reliability and availability of each peer in the database and calculates an average integer value. The addressing is realized as follows. If a peer P1 wishes to store a file in the cloud, it must submit a request to tracker, which selects the host peer. Peer P1 can receive an IP address to upload its file to the host peer by itself, or the file can be addressed by the tracker directly. Different host peers can have the same availability level. In this case, tracker selects peer with the largest storage space. Tracker application is composed by the following functions: receive, *exceed_client, reply_peer, reloj, tie_file_location, peer_space, dynamicity, file_list*. Flow diagram for tracker application (tracker.c) is shown in figure 5.

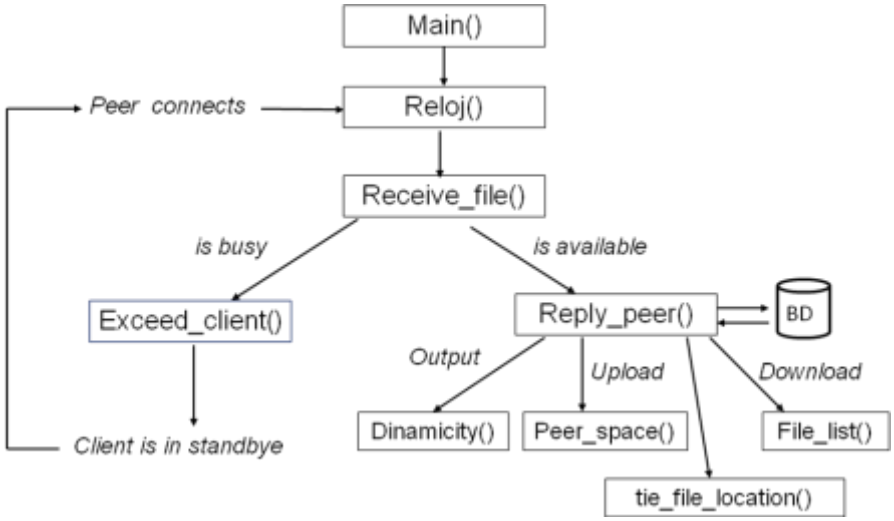

Fig. 5. Flow diagram for Tracker application

Database, is designed to register and control all information received by the tracker from the peers. The database in this cloud system is based on a relational model. In its design we consider three entities: reputation metrics, peers and files. Data recorded and monitored for each peer are: physic address, IP address, date and time of the last connection, available space in disk, number of rejections, availability, dynamicity, true and reliability. Data recorded for each file are its name, size and location. Files are localized using the peer's physical address. Relational model is useful to maintain a right relationship between file, proprietary and storage place.

5 Conclusions

During the last years the centralized cloud computing have evolved to a federated approach. In this scenario, P2P networks emerged as an ideal infrastructure to deploy cloud computing in the future. This paper proposes and implements a P2P cloud model considering different approaches such as reputation and collaborative storage. Compared with other storage models such as cloud storage based on client-server, distributed storage or P2P distributed storage, our proposed model offers different benefits such as scalability, confidentiality, file sharing, data replication, data management, quality of service, decentralization, and transparency. However, unlike the P2P distributed storage, our proposed model does not offer file fragmentation. We plan to integrate fragmentation and a distributed tracker in our model as future work.

References

1. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I. and Zanaria, I.: Above the Clouds: A Berkeley View of Cloud. In: Technical Report No. UCB/EECS-2009-28, University of California at Berkeley, Berkeley, CA, USA, (2009).
2. Mell, P. and Grance, T.: The NIST Definition of Cloud Computing. In: Special publication 800-145 (draft), Gaithersburg, MD, USA, (2011).
3. Lakshman, A. and Malik, P.: Cassandra - A Decentralized Structured Storage System. In: Workshop on Large-Scale Distribued Systems and Middleware (LADIS'09), Big Sky, MT, USA (2009).
4. Babaoglu, O., Marzolla, M. and Tamburini, M.. Design and implementation of a P2P cloud system. In: 27th Annual ACM Symposium on Applied Computing (SAC'12), pp 412–417, Trento, Italy (2012).
5. Sosinsky, B.: Cloud Computing Bible. In: Wiley Publishing Inc., Indianapolis, IN, USA (2011).
6. Cunsolo, V., Distefano, S., Puliafito, A. and Scarpa, M.: Cloud@home: Bridging the gap between volunteer and cloud computing. In: 5th International Conference on Emerging Intelligent Computing Technology and Applications (ICIC'09), Ulsan, South Korea, (2009).
7. Trajkovska, I., Salvachua Rodriguez, J., and Mozo Velasco, A.: A Novel P2P and Cloud Computing Hybrid Architecture for Multimedia Streaming with qos Cost Functions. In: ACM International Conference on Multimedia (MM'10), pp. 1227-1230, Firenze, Italy (2010).
8. Xu, K., Song, M., Zhang, X. and Song, J.: A Cloud Computing Platform based on P2P. In: IEEE International Symposium on IT in Medicine Education (ITME '09), pp. 427-432, Shandong, China, (2009).