Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingenieros de Telecomunicación

A Research Platform for Hybrid Models between Cloud Computing and Open Source Paradigms like P2P Streaming and Voluntary Computing with QoS Cost Functions

TRABAJO FIN DE MÁSTER

Irena Trajkovska

2010

Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingenieros de Telecomunicación

Máster Universitario en
Ingeniería de Redes y Servicios Telemáticos

TRABAJO FIN DE MÁSTER

A RESEARCH PLATFORM FOR HYBRID MODELS BETWEEN CLOUD COMPUTING AND OPEN SOURCE PARADIGMS LIKE P2P STREAMING AND VOLUNTARY COMPUTING WITH QOS COST FUNCTIONS

Autor
Irena Trajkovska

Director
Joaquín Salvachúa Rodríguez

Departamento de Ingeniería de Sistemas Telemáticos

2010

Abstract

The Peer-to-Peer (P2P) technology as a pioneer in the world of file sharing networks has made a remarkable progress attracting many followers during the last decades. Implemented as an organized overlay, it captivated the interest to design various protocols, topologies and novel applications based on this technique. Multimedia streaming has been highlighted ultimately as one of the common use cases built on the top of the overlay that significantly dominated over P2P file sharing market. This has pulled out additional issues on a lower network level causing problems that interfere with its accurate functioning. Network Address Translation (NAT), has been introduced as a temporal solution of the IPv4 addressing problem. This caused major interruptions to the P2P streaming communication and has entailed a lot of work to the engineers in order to find an appropriate solution for successfully traversing the NAT in an environment of P2P Multimedia scenarios. Soon after the market has nearly saturated of the non profitable P2P paradigm, a fast flesh back to the already familiar client-server model was on the way to threaten its existence. The paradigm of Cloud Computing was introduced and has immediately after start a fast conquer of the business market. Today, cloud computing offers different service models as a base for successful end user applications. Imposing the idea of "computing as an utility", it quickly attracted the attention of the dominant companies like Google, Amazon, Oracle, IBM etc, to completely change their vision of computing, migrating their existing platforms to build the cloud and redirect their power to host its users. In the same time, as a novel and still not explored area, it tickles the brain of many researches to focus on new ideas, designs, topologies, protocols and architectures, all addressed to the "cloud", promising a big revenue and interesting solutions for the computing of the future. However security is a crucial issue and main problem threatening the would of cloud. The problems of how to cope with heterogeneity of resources, manage the scalability of a platform that integrates the entire network topology (from the physical up to the application layer) and above all protect user data keeping its confidentiality, has raised to be the prime concerns and decision factors for many new users of the cloud models. Cloud computing has been compared and contrasted to other paradigms in order to suggest a plausible hybrid models that could take the advantage of the combined architectures. As such, volunteer computing has been observed in depth as another type of computing using a similar resources as the cloud model, but directed to other type of users - volunteers. A possibility to combine it with the cloud computing model has been researched in order to mutually increase their utilities.

This thesis contributes to the above mentioned paradigms, trying to investigate thoroughly each of them, on the way to define new ideas for possible architectures, protocols and solutions to problems such as NAT in P2P multimedia environment, Cloud security and find possibilities for synergy of the investigated technologies. Starting with the P2P paradigm and presentation of the existing P2P Streaming techniques it analyze the Quality of Service (QoS) issue as an essential part for this kind of networks. Skype VoIP client is examined, as a particular representative of the P2P applications, through comparative study of various research works and related analysis. The problem of NAT is observed with variety of techniques proposed for its successful traversal. More specifically some experiments have been executed within the environment of an existing project, in order to simulate the behaviour of a multimedia conversation in a scenario with applied NAT. Furthermore a research work of a distributed cloud computing network for multimedia streaming, in a both centralized and peer-to-peer distributed manner is presented, i.e., an attempt of joining the P2P and Cloud Computing paradigms. The suggested architecture merges private and public clouds and it is intended for a commercial use, but in the same time scalable to offer the possibility of non-profitable use. In order to take advantage of the cloud paradigm and make multimedia streaming more efficient, APIs in the cloud are introduced, containing build-in functions for automatic QoS calculation, which permits to negotiate QoS parameters such as bandwidth, jitter and latency, among a cloud service provider and its potential clients. Additionally cloud computing is accessed from security perspective, trying to understand and underline its importance, its establishment within the SLA, investigating it current status, the existing security protocols and various security issues and risks that threaten cloud third party users. At the end general perspective of voluntary com-

puting is examined mediate the existing and quite successful example projects in this field, to finally foresee another possibility of synergy this time between cloud computing and voluntary computing.

Resumen

La tecnología P2P (Peer-to-Peer) como pionera en el marco de intercambio de archivos ha logrado un notable éxito atrayendo un considerable número de seguidores durante las últimas décadas. Implementada como una "overlay" organizada, ha despertado el interés en numerosos profesionales para desarrollar varios protocolos, topologías y nuevas aplicaciones basadas en esta tecnología. La descarga multimedia ha sido últimamente sealada como una de las principales causas comúnmente empleadas en el mercado de intercambio de archivos P2P. Todo ello ha generado nuevas deficiencias o problemas en las redes de menor nivel, causando problemas que pueden afectar a su correcto funcionamiento. Network Address Translation (NAT), ha sido introducido como una solución temporal para el problema de direccionamiento del IPv4. Esto ha causado interrupciones constantes en la comunicació P2P streaming y consecuentemente ha obligado a los ingenieros a trabajar duro para buscar soluciones apropiadas para atraversar la NAT a escenarios multimedia de P2P. Inmediatamente después de que el mercado se hubiera saturado con paradigmas no útiles de P2P, una vuelta al modelo familiar de cliente-servidor estaba en marcha para amenazar su existencia. El paradigma de la Cloud Computing fue introducido e inmediatamente comenzó una rápida conquista del modelo de negocio. Hoy en día, Cloud computing ofrece diferentes modelos de sercicios como base de aplicaciones con éxito para usario final. Imponiendo la idea de "computing as an utility", rápidamente atrajo la atención de las compaías dominantes como Google, Amazon, Oracle, IBM etc, para modificar completamente su visión de computacion, migrando sus plataformas ya existentes para construir la nube y redireccionar su potencial hacia alojar a sus usuarios. Al mismo tiempo como área nueva y no explorada, ha estimulado el cerebro de muchos investigadores para centrarse en nuevas ideas, diseos, topologías, protocolos y arquitecturas, todo ello direccionado a la "nube", prometiendo un gran desarrollo en interesantes soluciones para la programación del maana. No obstante, la seguridad es uno de los principales problemas que amenaza a "la nube". El problema de como gestionar toda clase de recursos heterogéneos, dirigir la flexibilidad de una plataforma que integre por completo la topología (desde la capa fisica hasta la capa de aplicacion) y por encima de todo, proteger los datos del usuario manteniendo su confidencialidad, ha supuesto la preocupación más importante y ha decidido factores para tantos nuevos usuarios de los modelos "nube". La programación nube ha sido comparada y contrastada con otros paradigmas buscando un posible modelo híbrido que pudieran aprovechar las ventajas de las arquitecturas combinadas. De esta forma, la programación voluntaria ha sido obsevada en profundidad como otro tipo de programación usando recursos similares al del modelo "nube", pero dirigido a otro tipo de usuarios-voluntarios. La posibilidad de combinarla con la computaición del modelo nube ha sido investigada con el propósito de incrementar sus beneficios.

Esta tesis contribuye con los paradigmas mencionados anteriormente, buscando investigar a través de cada uno de ellos, en el camino para definir nuevas ideas para posibles arquitecturas, protocolos y soluciones a problemas tales como NAT en el entotorno de P2P multimedia, la seguridad de la nube y encontrar posibles sinergias entre las tecnologías estudiadas. Comenzando con el paradigma del P2P y las existentes técnicas de PSP Streaming, analiza el problema de la Calidad del Servicio (QoS) como una parte fundamental de sus redes de trabajo. Skype VoIP client es examinado, como ejemplo representativo de las aplicaciones P2P, a través del estudio comparativo de varios trabajos de investigación y análisis vinculados. El problema del NAT es observado a través de una variedad de técnicas propuestas para suéxito. Más específicamente, algunos experimentos han sido ejecutados sin el entorno de un proyecto pre-existente, con el propósito de simular el comportamiento de una conversación multimedia en un escenario con un NAT entroducido. Posteriormente, se presenta un trabajo de investigación acerca de una red distribuida "en nube" para el streaming multimedia, en una manera centralizada y peer-to-peer distribuida como un intento de unir el P2P y los paradigmas de la programación "en nube". La arquitectura sugerida emplea tanto nubes privadas como públicas y se espera de ella un uso comercial, pero al mismo tiempo es lo suficientemente flexible como para ofrecer la posibilidad de un uso no lucrativo. Con el propósito de usar las ventajas del paradigma de la "nube" y lograr que el streaming multimedia sea más eficiente, se han introducido APIs en la nube, conteniendo funciones integradas para el cálculo automático de QoS, lo que permite negociar y establecer parámetros como el ancho de banda, jitter y la latencia, entre un proveedor de servicios "nube" y sus potenciales clientes. Adicionalmente, la programación "en nube" es observada desde la perspectiva de la seguridad, tratando de comprender y subrayar su importancia, su establecimiento con el SLA, investigando su status ctual, la existencia de prtocolos de seguridad y varios problemas de seguridad y riesgos que amenazan a terceras partes. Al final, la perspectiva general de la programación voluntaria es examinada mediante la existencia de bastantes y exitosos ejemplos de proyectos en este campo, y finalmente predecir o intuir otra clase de sinergias, esta vez entre programación "en nube" y programación voluntaria.

Contents

List of Figures

Chapter 1

Introduction

1.1 Key Research Issues

Within the last decades the Internet has hosted increasing number of applications for direct sharing of files among different users without the need of a server as a mediator. This peer-to-peer manner has distinguished remarkably form its predecessor, the client-server model of communication. Offering number of applications for file exchange this model is organized at the top of the network layer in a so called overlay network. Independent from the network level protocols it soon started to build it own model of communication among the participants based on protocols specially designed for its purpose.

As a quite independent structure, it attracted the attention of the Internet community to take this advantage as a challenge for investigating various applications for improved P2P exchange of most of all, multimedia files. The characteristic of scalability and high availability of the shared resources is what was crucial factor for this architecture to sustain the many legacy attacks following its development. The massive acceptance and approval by most of the users on the Internet is another key in its favour.

After reaching the top popularity mediate BitTorrent as the leading P2P application, the focus was turned toward creation of application for streaming of multimedia files. P2P streaming is today one of the two types of video distribution besides video on demand which reaches high utilization thanks to the existing services and applications that makes this feasible to increasing number of users.

However apart form the proven advantages an overlay architecture faces with many drawbacks, most of all due to the lack of communication with the network layer and thus the inability to make an accurate and precise peer selection. This is what prevents the P2P model to increase and be developed for a further commercial purposes, but this same flaw was considered latter as a motivation in order to get closer the Internet service providers (ISPs) and the P2P users on the way for better network traffic and resources organization.

Quality of Service (QoS) has been considered a very important mechanism for the control degree of the reserved resources. As the most significant QoSs parameters in the area of P2P multimedia communication can be pointed jitter, bandwidth, latency, round-trip-time, etc. Various approaches exist today in the academy world, for increased QoS control by offering scientific techniques for their measurement in the networks such as P2P which makes it difficult due to the dynamic nature of the P2P overlay network.

Another problem that seriously threaten the end-to-end communication is the Network Address Translation method (NAT), which was initially introduced as a temporal solution for the problem of the IPv4 protocol in order to deal with the exhausting address space. Solving one, NAT has opened another problem becoming the biggest enemy for all the types of communication with established NAT mechanism on the way in between their participants. In the case of P2P concretely, it was impossibly at the begging to establish a direct connection if some of the participants was behind a NAT. Latter techniques has shown a big achievement related to this issue, offering many research proven ideas. Nevertheless, while particular solutions may help resolving concrete problems, the question of design a global NAT traversing technique stays still open.

The initial focus during the work of this thesis was oriented towards the P2P architecture and further knowledge of the existing state of the art in this area. It resulted that as a quite mature area had a lot to offer, beginning form concrete applications to many through analysis and suggestions. The issue of the NAT was successfully studied mediate a small experiment simulation where the possible interference of the NAT in a multimedia scenario were investigated. The objective related to this work, was to analyze the behaviour of a P2P multimedia streaming applications and how seriously the introduction of NAT can affect and jeopardize the communication.

Further work oriented the interest of a possible application of the P2P technology in another domain that way constructing a hybrid model. This model would intent to provide a common utility for the both paradigms and take the advantage each of them offers.

Cloud Computing as a novel paradigm has filled this gap enabling completely new approach and vision in the world of computing. As a paradigm being still in the evolving phase, there is no commonly excepted definition for what exactly cloud computing stands for. A generally accepted view for the cloud computing identifies it as ”computing as a utility”, a complete new approach towards a complex utility that includes applications as a services distributed over the Internet together with the hardware and system software in the data centers that provide those services.

Offering a complete platform from the lower physical layer to the end-user application layer, created for deployment of different type of services, cloud computing is just at the beginning of its evolution. This opens the door for many new research issues and business concerns to be set up as a goal objectives to build up a more advanced platform and services withdrawing the best preferences of the offered tools and utilities.

Supported and reinforced by the strong lobby of leading companies as Google, Microsoft, IBM, Amazon, etc. the cloud model has put important emphasis on the market effect and its possible business models. By classifying the services in the three basic models (IaaS, PaaS, SaaS), the idea is to get it closer to different category of users, i.e., from a particular individuals, small and medium-sized companies, until the big companies and institutes. Private and public cloud can be chosen as an optional models on the way of establishing the virtualization of the resources.

The advantage of being novel and not explored area, bring an additional concern related to standardization and definite determination of the cloud model. There are still not well defined and generally accepted protocols especially in the area of security. As a relevant problem, security does not permit an improvisation of a temporal methods and solutions, due to the complexity of the architecture and the user data confidentiality stored over thousands of clusters whose geographical location is omitted form the user. The benefits of the reserve resources and virtual environment could result in serious failures if not approached with appropriate methods and dealt in a suitable way, with respect to the mutually established SLAs and QoS among the cloud providers and cloud users.

1.2 Motivation

Entering further in assessment of the QoS properties in P2P systems and different ways to measure, predict and include them in a possible price models, was the diver initiative of investigation and short term goal in the entire work. Moreover, the possible integration of a P2P within the cloud model, by examining the different opportunities to enable correct and accurate services to the end-users, but in the same time, profit to the providers, was an additional idea that could easily fit in the whole scenario.

The related researched work treats in details the distribution trees topology, numbering various use cases of how to benefit from the cloud. However a lack of a concrete solutions in the domain of QoS guarantees in streaming scenarios was noticed. Therefore as principal objective in the work, was considered the creation of automatic API functions based on QoS parameters to leverage a novel hybrid cloud and P2P architecture for multimedia streaming. Cloud paradigm is favoured to be future key for bringing back the Client-Server (CS) topology in multimedia communication, and by expanding it to support P2P streaming there is a possibility it can bring double benefit to both the cloud service providers, and the end users.

Furthermore, the existence and successful results of another analogical paradigm to the cloud computing, voluntary computing, has fit the research attempt of finding out all possible applications of the cloud model in another similar domain. Not only application, but rather, synergy could be achieved between the cloud and volunteering model, as the academy world have already proven it feasible.

For the moment my focus is concentrated toward profound analyses and simulation of the proposed hybrid cloud and P2P model, with possible implementation for industrial causes, nevertheless starting from the origin of their nature, it is not excluded the possibility new models to emerge between the cloud computing and voluntary computing as well.

1.3 Organization of the Thesis

This thesis is organized as follows: Chapter 2 presents a survey of the various P2P streaming architectures. We discusses Quality of Service and their influence in the P2P streaming architecture, presenting on the way example solutions for QoS control and calculation. Skype is selected as a specific P2P representative based on a VoIP technology, to be discussed and compared to a vide range of related researches. NAT technique is described in Chapter 3 and various technique putting stress on the existing models for traversing NAT in P2P application scenarios. As example a simulated multimedia scenario followed, including the conclusion and results of the experiment.

In Chapter 4 Cloud computing is defined and various models are presented. We talk about the business aspect of the cloud model, popular cloud platforms are discussed distinguishing the most remarkable challenges of the cloud paradigm. Moreover it includes cloud computing security requirements and discusses possible solutions and methods to deal with it, pointing out the risks, security issues and standard security procedures that cloud computing has to face during its development. Chapter 5 represents an overview of the suggested architecture and description of the API functions and calculation of QoS parameters. In continuation Chapter 6 describes the volunteer computing from its origins through various characteristics, selecting the Boinc project as a most remarkable in this domain. Chapter 7 discuss the possible hybrid architecture of both cloud- and voluntary-computing presenting such an example model. Finally, Chapter 8 concludes this thesis and outlines some future work in this area.

Chapter 2

P2P Technology

2.1 Introduction to P2P

After the first appearance, **Peer-to-Peer (P2P)** technology has undergone huge popularity, while spotting exponential raise. Developed over the years, in order to overcome the client-server based limitations, It attracted thousands of developers to create applications for P2P content exchange, opening in parallel many research gaps and offering variety of ways to resolve them.

Bit torrent (62) as a pioneer in this area, took remarkable place in the development, raising the battle for variety of offers in the world of the P2P applications. Currently is still considered to be leader among the P2P file-sharing applications.

Soon after P2P file sharing market started to saturate, a new way of distributing the content was introduced - *multimedia streaming.* Since the traditional client-server based video streaming solutions incurred expensive bandwidth provision cost on the server, the P2P paradigm has been implemented in such a way to deliver the content to the audience as an alternative way to the traditional client-server model. In addition, various researches have proven this new model to contribute in increasing the robustness and scalability. Therefore, it could be considered that Peer-to-Peer (P2P) represents a new standard for building distributed network applications.

So far, there have been designed various P2P streaming systems. They provide two different types of multimedia video distribution: **Video-On-Demand (VoD)** and **Live Streaming** services on the Internet at low server cost. Consequently this attracted many users to switch to the revolutionary way of watching media content without the need to be download.

Build as an overlay network consisting of various peers, P2P video streaming has received a lot of attention, as it appears as an alternative to the conventional IPTV systems. P2P TV has evolved to be a cutting edge technology for delivering multimedia streaming to the public over the Internet which provides good enough quality to imitate the IPTV.

However, an overlay architecture, despite its advantages, faces with different problems arising from the network-unaware peer selection. As peer number increases, so does the damage this problem causes. In order to reduce network-oblivious traffic, various P2P model based on collaboration between ISP and P2P users have been recently suggested. Reducing traffic, contribute to improvement of network performance, but does not necessarily guarantee optimum performance experienced by the users of the real-time streaming service.

Quality of service (QoS) control issue is very important to be taken into account in streaming multimedia systems. Most of the application in order to support effective real-time IPTV services require (QoS) guarantees. Most of those include minimum path bandwidth, jitter and bounded end-to-end latency for the delivery of their data streams. Traditional connection-oriented routing schemes for multimedia data streaming assume single-path routing, and often attempt to find the best path based on one or two QoS metrics. From the viewpoint of a distributed multimedia application, a path is useful only when it can satisfy all the QoS requirements.

Different architectures have been developed as a possible way of relaying the content to the end-users. Generally they are divided as **Tree** and **Mesh** based approaches in P2P networks. They will be described in details in the following section.

2.2 Survey of the Peer to Peer Streaming Techniques

Video streaming is divided in two categories:

- **Live**. In a live streaming session, live video content is sent in real-time to its users. Synchronization exist among all video playbacks.
- **On-demand**. Video-on-demand users enjoy the flexibility of watching chosen video clips at any time they desire. There is no synchronization among the playbacks of the same video clip on different users.

On the other hand, based on the overlay network structure, P2P streaming systems can be broadly classified in two categories: **Tree-based** and **Mesh-based**. This section, introduces three P2P live streaming systems using different overlay structures.

2.2.1 Tree-based P2P Streaming

Tree-based P2P Streaming represents an application level structure containing multiple trees, that is formed among the participating peers in a streaming session, just like IP multicast tree that is formed by the routers at a network level. This tree is rooted at the video source server such that the overlay dictates a deterministic route for each data block to traverse from the source to its destination.

After creation of the overlay, each user joins the tree at a certain level. The paths that the blocks should traverse during the delivery are determined by the overlay. Users receives the video from their parent peer at the above level and push the received video data to their children peers at the below level. Depending on their access link bandwidth, each peer determines number of trees to join.

The tree based overlay should be able to adapt to any network condition changes in order to adjust the block distribution path and obtain an optimal efficiency.

MutualCast (86) excepts this rule and achieves reliable content delivery by re-adjusting the delivery tree rate according to the change. This is done by using a dense distribution forest. However the large number of nodes makes it difficult to scale. Early examples of single-tree based streaming include Overcast (88) and ESM (60).

Figure 2.1, depicts an example of two streaming delivery trees.

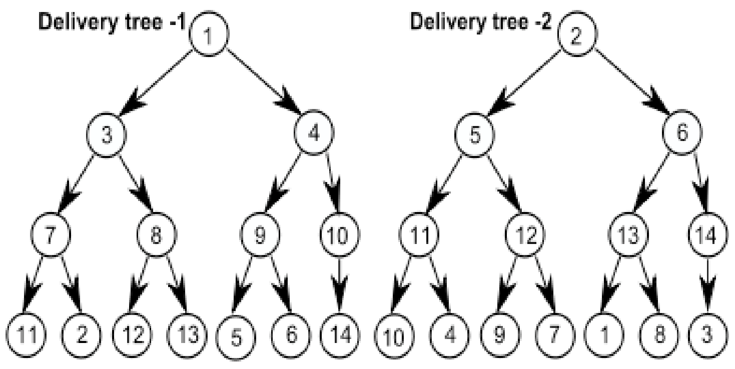


Figure 2.1: Delivery trees in Tree-based approach

Most important advantage of the tree-based organization lies in the reduced latency during the process of content delivery, seen in the facility of every intermediate node to forward immediately to its successors the received data block . As a major drawback is the lack of robustness to peer churn. A peer departure will temporarily disrupt video delivery to all peers in the sub-tree rooted at the departed peer.

In order to decrease the churn and increase the efficient use of available system resources, peers can be organized into multiple diverse trees. To that end, every peer is attached as an internal node in one tree, and as an external node in the rest of the trees. Afterwards, every peace of encoded content is delivered through a specific tree, mediate simple push mechanism. Hence, as most important in tree-based P2P streaming approach is the algorithm for tree construction (105).

2.2.2 Multi-tree Streaming

Multi-tree streaming, is approach where the server divides the stream into multiple sub-streams, thus obtaining multiple sub-trees, one for each sub-stream, instead of one streaming tree. For locating the sub-streams, each peer should join all sub-trees. In the frames of every sub-tree, the associate sub-stream flows down from the source server to the lief nodes, level by level. This approach was suggested as a consequence to the leaf nodes problem, i.e., unused bandwidth resources of the leaf peers, when redistributing streaming content, due to the architectural luck of the tree-based topology. Typical example approaches of this type can include SplitStream (57) and Bullet (94) architectures.

The idea of multi-tree streaming organization is as fallows. A peer can be positioned in different sub-trees, as an internal node in one sub-tree and on as a leaf node in another sub-tree. The key feature is the peer's uploading bandwidth that has to be use in order to upload a sub-stream any time the peer finds itself to be an internal node of a sub-tree. To achieve high bandwidth use, the number of sub-trees in which a peer is placed on an internal node can be set to be proportional to its uploading bandwidth (101). On Figure 2.2, is shown an example of such multi-tree sub-stream delivery trees.

2.2.3 Mesh-based P2P Streaming

In the mesh-based approach, rather then fallowing a static topology, the participating peers form a randomly connected overlay, called mesh. The delivery path of the data blocks is based upon not only the overlay, but also on the feedback from the neighboring peers. The establishment of the overlay depends on availability of the content and the bandwidth of the peers.

Every peer connects to a subset of random peers maintaining a certain number of parents (incoming degree) and number of child peers (outgoing degree). It then receives status information from its immediate neighbors and makes a distributed decision for content flow in the overlay. Video content is pulled by a peer from its

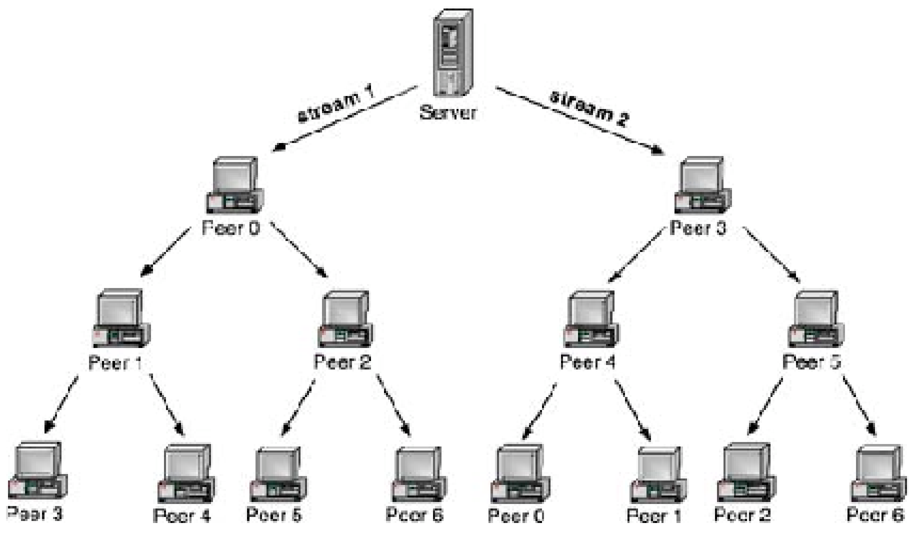


Figure 2.2: Multi-tree based streaming with two sub-streams and seven peers

neighbors who are already containing the content. The same status information continues to be received also during the content distribution.

The information for outgoing degree of all nodes can be contained in the same time by so-called *bootstrapping node*. It selects random subset of peers that can accommodate a new child peers upon their arrival in response to an incoming request for parents.

Pair-wise connections in the mesh-based approach can be used for content delivery in both bidirectional or unidirectional fashion.

It can be foreseen due to its structure, that mesh-based delivery is amenable for creation of an over redundant overlay. Most famous implementation of mesh-based delivery applications are BitTorrent (62) and PPLive (30).

Mesh-based video streaming show to be very robust to peer churns, due to the steadily maintained connections among the multiple neighbors. Unfortunately the high degree of dynamics in the system makes the video distribution be unpredictable. It is uncertain the way that data packets traverse on their way to users. Furthermore, the mesh-based delivery has to delay the forward of the received block until status information has been gathered, to make sure no duplicate delivery is performed. Consequently, users may suffer from video playback quality degradation ranging from low video bit rates, long start-up delays, to frequent playback freezes (105). Organized view of random mesh overlay in details can be observed from the Figure 2.3.

The key difference between the mesh-based and the tree-based approaches is how the delivery tree of individual packet is formed. In addition, mesh-based delivery show to be more robust to churn, whereas the tree-based proves to reduce latency when delivering the content.

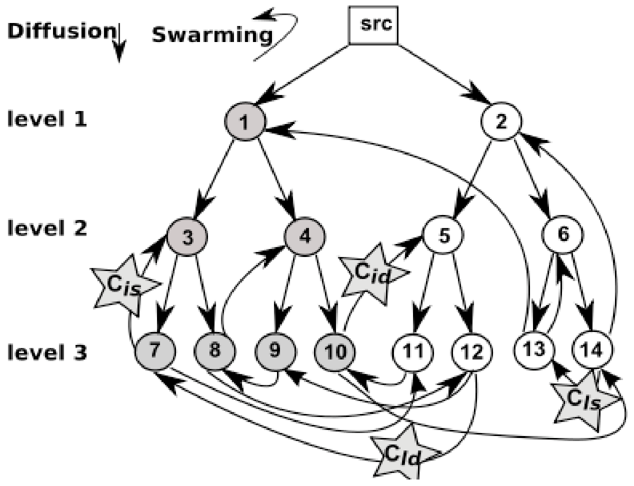


Figure 2.3: Organized view of a random mesh

The tree based overlay should adapt to any network condition changes in order to adjust block distribution path and obtain optimal efficiency, while mesh-based delivery organizes the data flows according to the status information, which incurs easier implementation of the second one (104).

2.3 QoS in P2P Multimedia Streaming Systems

Quality of Service (QoS) is a guarantee measurement of the performance level, that is supposed to serve the end-user application or, it is a measurement of how well the network operates and a means to define the characteristics of specific network services. The Internet as a best effort network, is not able to guarantee the user, a certain level of performance they can expect from the network.

Video streaming and Video on Demand (VoD) as real-time applications, offer different advantages such as scalability and availability. However due to their dynamics, they happen to be particularly sensitive to QoS as they can suffer from peers' churn, congestion, and links failure as well as occasional problem caused by the limited supplier peers on the overlay. Therefore, P2P streaming applications would not tolerate various fluctuations in network. To that end, the QoS mechanism should guarantee continuous content delivery through P2P overlay. Since in a P2P network, no peer entity has control over the entire network, the guarantee of QoS must be interpreted statistically.

Incorporation of an appropriate admission control strategy can be helpful to guarantee an acceptable level of QoS for real-time content delivery over P2P networks where a single peer seeks to receive video content from multiple sender peers having distinct characteristics (111).

In order to coordinate better the process of upload and download, peers offer their bandwidth and memory to interconnect far until the end-users. This solution avoid the "unfairness" between peer participants, i.e., continuous exhaustion of specific peers' bandwidth. However it brings consequences on the Quality of Service (QoS) and furthermore, it decreases the bandwidth and increases the latency which can drastically affect the Quality of Experience (QoE). On the other hand, QoS is essential for P2P streaming, but should not permit side effects like load balancing and locality awareness (46).

As mentioned earlier, P2P streaming systems represent a distributed, dynamic network consisting of large number of heterogeneous peers. Such systems require scalability in order to serve the large number of peers in a term of accuracy, convergence speed, and extra overhead per peer. Balancing all QoS is very important issue which especially affect the scalability of the system. Moreover the entire maintenance has to rely on secure mechanism which should keep up the entire integrity of the network.

The article *"Scalability of Peer-to-Peer Systems"* (107) elaborates scalability profoundly, as a comparative study in existing systems such as Gnutella, Tapestry, Pastry and Chord.

Fallowing is an elaboration and some example research works of the most significant QoS in P2P streaming architecture.

2.3.1 Latency

Latency measures the delay in a P2P system, i.e. or the time one data packet need to pass from the sending peer to the receiving peer mediate the P2P overlay. If the delay between two peers takes longer then tolerated, the protocol loses its sensitivity to react to the short term network dynamics, since there is no fast enough feedback. In P2P latency is measured by Round-Trip-Time (RTT), between two peers reflected by the "ping" time.

Jin Li in (99) presents an example tool called **Virtual coordinate system** for latency estimation.

In (138) Dongyan Xu et al. propose an algorithm **OTSp2p** to be executed by the requesting peer, before the P2P streaming session starts. This algorithm computes the shortest buffering delay through finding out what is the optimal transmission schedule to coordinate a given set of supplying peers.

Another interesting solution creates an **adaptive video streaming mechanism** by organizing the participating peers in clustered overlay networks in a shape of a small world. The simulation results prove remarkable improvements in the received video quality, resulting in improved packet delivery and reduced latency, until an overall QoS guarantee for the video content (44).

Stanford P2P Multicast streaming system (52) employs overlay architecture specifically designed for low delay video applications. This system keeps the end-to-end delay as small as possible and guarantees high video quality.

Similarly, Eric Setton et al. from Stanford University invent a **distributed streaming protocol** to enable P2P multicast streaming. This work introduces an adaptive scheduling algorithm that offers video streaming to a large population of peers with low latency and high scalability (124).

2.3.2 Jitter

Jitter is a QoS of the end-to-end transit variation delay of packets in a given stream. Similar to client/server based video streaming systems, the primary QoS metric for P2P video streaming systems is reflected by the jitter rate. Ideally, packets should be delivered in a perfectly periodic fashion. Even though the transmition uses evenly spaced stream, jitter is unavoidable due to the variable queuing and propagation delays. This provokes packets arrival in a wide range of inter-arrival times to the destination. The jitter also appears as a result of conflicts with other packets who use the same links, and non deterministic propagation delay in the data-link layer.

When watching a video stream from the Internet, users can tolerate a small and variable amount of initial buffer time. On the other hand, once playback starts, users are far less willing to tolerate jitter. In a P2P network, no peer entity has control over the entire network. Therefore, guarantee of QoS must be interpreted statistically.

The traditional approach to analyzing jitter rate is to directly study the download rate experienced by peers. This posts a major challenge to P2P video streaming network designer because a P2P network is essentially a large number of peers bound together by a common protocol stack. While network designers attempt to regulate peer behavior in their programs, maintaining the integrity of the original code is extremely difficult. As a result, there is no guarantee that a connected peer is running the original program.

Exist, apart from the common approach, particular solutions, like designing a successful P2P network with trying to limit jitter rate during client playback with high probability.

Another, apply some sort of buffer-and-play mechanisms i.e., starting to display the video only after it is fully downloaded. By doing this, jitter is 100% avoidable, but it also results in the longest initial delay and the maximal buffer size, which is generally unacceptable, especially for small wireless devices.

Xiaosong Lou et al.in *"Quality of Service in Peer-to-Peer IPTV Networks"* (103) demonstrate **different approach**. They express jitter by studying the distribution of peer download latency, instead of measuring peer download rate. Based on the distribution, the upper bound of a jitter rate is derived. It is considered that the upper bound not only helps estimating the QoS of an existing P2P IPTV network, but also serve as a guideline for designing new P2P IPTV networks.

Another jitter based article (106), studies **simple distributed model** for jitter control through algorithms measured in a term of guarantees relative to the best possible by an off-line algorithm. According to the results for delay jitter, simple algorithm of filling half the buffer has very strong relative property. They purpose simple algorithm for rate jitter, with release rate proportional to the fill level of the buffer.

2.3.3 Bandwidth

Bandwidth is a measurement of the rate at which data can be sent through the network. It is measured in bits per second (bps).

In P2P streaming applications this QoS is essential factor in the intercommunication and the delivery of a data stream. The dynamic formation of the P2P links, can introduce consequences like formation of bottlenecks, as a result of the fluctuate bandwidth along the network. Bandwidth is a measurement that can not be assumed or calculated with certainty. The frequent peer joining and leaving, increases the difficulty of getting a comparatively similar bandwidth between peer pairs. Fortunately there exist methods that can proceed network bandwidth oriented approximation and adaptation.

In the paper (140), is proposed an algorithm called **TOBAB (Trend-Oriented Bandwidth Adaptive Buffering)**. It adjusts the request quantity sent to each source peer. This algorithm does not rely on accurate bandwidth value between two peers. Instead, it sends adaptive requests (tread on the heels of the bandwidth trend) and greedy (take full advantage of available bandwidth).

2.3.4 Robustness to Malicious Peers

System that calculates the global reputation score of a peer by considering the opinions (i.e., feedbacks) from all other peers who have interacted with this peer, is called **reputation system**. These systems usually involve various evaluations as a feedback, after some streaming session by a certain peer, to be able to serve to other peers. This implies publicly available scores that will keep all peers informed about trustworthy and malicious peers.

The system should be robust to various attacks by both independent and collective malicious peers. Identifying trustworthy peers is especially necessary in commercial P2P applications, such as P2P auctions, trusted content delivery, pay-per-transaction, and P2P service discovery. Not less importance it also takes in the P2P streaming and file sharing applications.

PowerTrust (142) is a trust overlay network (TON) that models the local trust and reveal feedback relationship among peers. Their creators argue that the eBay user behavior is decentralized by nature, as the peers are autonomous entities who make decisions individually.

2.3.5 Awareness

Awareness enables to a peer detect and connect to the nearby peers and, in turn, contribute to maintaining low, overall end-to-end delay.

Related to the locality-awareness principle, there exist one interesting proposition by the authors Alhaisoni M el al. (46). For achieving locality awareness to the peers, they introduce a **cross-layer approach**, already used in TCP and QoS-enabled networks, between the overlay network and the underlay network. Monitoring technique allows gathering information from the participant peers of the underlay network. As parameter chosen to periodically provide up-to-date information it has been used RTT (round trip time), which reflects the status (latency) of the local peers to the individual nodes. It is achieved a contribution with this idea, to maintaining relatively low jitter. Moreover, this technique insures that peers had been able to connect to the closest nodes.

2.3.6 Churn and Congestion

Peer joins and leaves an open P2P system dynamically. Therefore churn is used to denote the continuous process of incoming and outgoing peer nodes into the overlay network. This of course is considered as undesired process since it directly infringe the overlay integrity and violates the other QoS metrics. The system should adapt to such peer dynamics instead of relying on predetermined peers.

The proposed algorithm in (46), apart form the other roles, deals also with controlling the peer churn and congestion. Participants' nodes status is checked periodically. This is done to detect congested peers and switch away from those. Peers having lower mutual RTT are then chosen as new stream source, realizing this way the concept of locality. Secondly, when a peer leaves the network, this can be detected because no response to the RTT request comes back. This will immediately trigger a forces handover to another peer, chosen among the available ones and having the lower RTT value. The handover period does not result in an interruption because the streaming process is supported by a redundancy degree. In this way, the QoS (and QoE) can be maintained stable.

Finally, there exist more examples on related researches, where various solutions for increasing and examining the QoS are presented. One good example of general QoS maintenance is the **PROMISE** solution (80). Answer is presented to the fallowing remarkable challenge: *"in a highly diverse and dynamic P2P network, how to select, monitor and possibly switch sending peers for each P2P streaming session, so that the best possible streaming quality can be maintained?"*

2.4 Skype as a Particular Type of P2P System for VoIP

Despite the typical P2P architectures that serves generally for file sharing and multimedia streaming applications, exist another type of implementation of this architecture. Skype is one example of application built on the top of the P2P overlay integrating the VoIP protocol that shows slightly different level of use. Moreover, Skype uses its own proprietary protocol in order to obtain confidentiality of its calls.

Apart from providing file-transfer services and Instant Messaging, Skype's basic use is the Voice over Internet Protocol (VoIP) feature. In fact it can be said that Skype represents a VoIP system build using the P2P architecture. More specifically, Skype underneath lies on "Supernode-based" hierarchical peer-to-peer overlay network, as shown in Figure 2.4. The super-node overlay forms the core of the P2P network. Multiple clients connect to the super-node and form the client overlay.

Skype dates back since August 2003 and was founded by Niklas Zennstrom and Janus Friis, the founders of **FastTrack** and **Kazaa**. Since the first appearance, it has shown to be extremely popular and useful application, whose interest increased exponentially throughout the years.

There are various developed solutions related to Skype and similar VoIP systems, but what makes Skype especially attractive is being a pioneer in the P2P

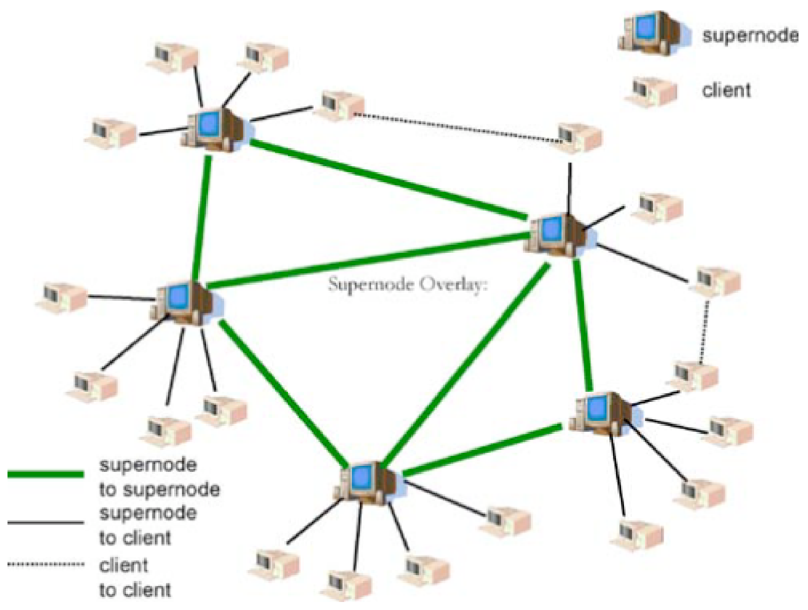


Figure 2.4: Supernode-based P2P overlay used in Skype

architecture based idea for VoIP system, i.e., Skype is the first VoIP client based on P2P technology.

2.4.1 Related Researchers

Currently, there are, many researches related to Skype. Some of them are comparative studies with similar VoIP systems, some base on analysis of the traffic and telephony protocol used, other focus on Skype's security issue, quantifying user satisfaction, etc.

For example (78) deals with understanding how P2P VoIP traffic in Skype differs from traffic in P2P file-sharing networks and from traffic in traditional voice-communication networks. It makes a simulating environment as a test-bed for examining Skype's traffic behavior by observing its calls during limited period of time.

An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol (43) is also of the initial articles related to Skype analysis. It treats Skype's key functions such as login, NAT and firewall traversal, call establishment, media transfer, codecs, and conferencing under three different network set-ups, by profound study of Skype network traffic.

There are several interesting outcomes form this paper. Some of them claim existence of **Skype Login Server** as the only central entity in Skype network, that stores user names and passwords of other users. Again it is supported the idea of the Super nodes and Ordinary hosts in a **"Super-node" network**. Moreover they believe in the existence of host cache (HC) table with IP addresses and port number of the super nodes.

Similar to the previous research paper, they claim no existence of global traversal server for NATs and firewalls, rather various techniques used for that purpose, such as STUN (117). Skype Client (SC) must establish TCP connection with Super Node (SN) (with public IP at port number 80 or 443). In their experiments the authors have come to conclusion that the login server is hosted by an ISP that is located in Denmark.

Briefly further on, there exist **Bootstrap super nodes**, SC uses login algorithm to determine available peers out of the nodes registered in the HC, call signaling is always carried over TCP, there is no support for silence suppression in Skype and it does not support full mesh conferencing (97). It showed to have reasonable quality of calls, enabling users to connect simultaneously from various computers.

Another interesting research *"Comparative analysis of the VoIP Software"* (66) is related to analyzing Skype performance and its comparison with Windows live Messenger and Yahoo Messenger. Results shows that Skype has the longest setup delay. WLM seems to show most quality when question about jitter, whiles YM shows to have the longest end-to-end delay.

Experimental analysis implementing similar analysis, can be found in (100), this time comparing Skype to MSN Messenger in the area of presence, connection setup delay, mouth-to-ear (M2E) delay, Mean Option Source (MOS) and Handover Support. Generally it was found out that Skype shows good voice quality but no better than MSN Messenger. It shows slightly longer delay than MSN Messenger which is tolerable. The P2P technique, enables conversation set-up regardless peers location, because it can work almost seamlessly behind NATs and firewalls.

The article (99) makes very thorough analysis on number of P2P applications (sharing and streaming), VoIP systems that include Skype, P2P scheduling, finally examining the different QoS denoted as measures of heterogeneity and proximity.

"VoIP and Skype Security, Simson L. Garfin" (74) initially compares Skype with other VoIP systems, ISDN, contrasts Skype to P2P, KaZaA and its performance over Dial-Up. Furthermore it reveals Skype's security issue through observation of Privacy, Authenticity, Availability, Survivability, Resilience, Integrity (Conversation), Integrity (System).

The overall observation says that *"Skype appears to offer significantly more security than conventional analogue or ISDN voice communications, but less secu-*

rity than VoIP systems running over virtual private networks (VPNs). It is likely that the Skype system could be compromised by an exceedingly capable engineer with experience in reverse engineering, or by a suitably-motivated insider."

As final example, in the area of QoE, I sort out (58). This work, proposes a model, specific to Skype, but generalized to other VoIP services, to quantify the VoIP user satisfaction. It is based on a rigorous analysis of the call duration from actual Skype traces, deriving an User Satisfaction Index (USI) from the model.

Exploring the user satisfaction is very important for the development of QoS-sensitive applications. The purposed solution, shown to be very advanced measurement in the area of QoS and QoE shown by the users.

Chapter 3

Network Address Translation in Multimedia Communication Systems

3.1 Introduction

The role of Network Address Translation in multimedia environments, has been topic in a number of research articles about New Generation Network and main problem in many related project. When NAT was initially introduced, there weren't taken into consideration all applicable affects it may cause on the way while still facing its utility. However, although it emergently saved Internet world, IPv6 seemed to threatened significantly the NAT's further necessity. Raised up with the popular growth of many multimedia topologies, the battle against NAT becomes more highlighted, defining generally two points of view.

The first considers avoiding NAT in multimedia applications where this is possible and doesn't interfere security issues, whiles second one, aims on trying to apply a suitable NAT Traversal method in such a scenario. Both considerations could bring suitable results and solve particular problems. While the first one doesn't seems to be always applicable, the second one, depending of the proper choice could resolve NAT problems up to a descent level.

Beginning with brief introduction of the NAT functionality, I will go further in this section, presenting, many possible existing techniques for its traversal.

3.1.1 Network Address Translation

Network Address Translation, (NAT) was defined originally within RFC 1631 in order to map IP addresses from one network to another, that way bridging the local network with the outer world. It allows single IP address to route entire local network to the public network, or when connecting multiple computers on the Internet it helps to conserve IP address. By using private IP addresses, a network only needs only one or very few public IP addresses for external communication.

NAT routes the communication through gateway server that forwards it on to the destination. By changing the source private IP address and port number to public ones for outgoing connection requests, NAT creates mapping to allow return packets to reach the original sender. Communication outside - inside on a local network containing NAT, first reaches NAT, which in turn performs a look-up of the destination address in its address table, and routes the data through the computer. One advantage of NAT, is that neither the internal machine nor the Internet host is aware of these translation steps.

Unfortunately, the protocols used for communicating rich media over IP networks such as H.323, SIP, MGCP and H.248 conflict with the most network security mechanisms like NAT (75). In the case of multimedia applications, the internal machine doesn't know that it's behind a NAT, as it only looks at the IP headers. Additionally, as the NAT is acting as a firewall, it makes an entry in its routing table to ensure that the return packets are passed through the firewall and are not blocked. For example, it might replace the source IP address with its external address and replacing the source port with a dynamically assigned port number (this port number is dynamically assigned by the NAT to be used for traffic from this internal host to the destination) (137).

The addressing table of NAT is stored in Dynamic Random Access Memory, DRAM. The size of this memory mainly determines the NAT's speed of operation. Theoretically, the size of an address table is about 160 bytes and for NAT, which typically has a 4MB DRAM, it could process about 26,214 simultaneous look-ups. This is more then satisfactory for most applications.

3.1.1.1 General Classification

Generally there exist two types of NAT, *static* NAT and *dynamic* NAT.

Static NAT does mapping of an unregistered IP address to a registered IP address on a one-to-one basis. Particularly useful when a device needs to be accessible from outside the network.

Dynamic NAT, on the other hand, maps an unregistered IP address to a registered IP address from a group of registered IP addresses. Dynamic NAT also establishes one-to-one mapping between unregistered and registered IP address, but the mapping could vary depending on the registered address available in the pool, at the time of communication (61).

Network Address Port Translation, (NAPT) is a method by which many network addresses and their TCP/UDP ports are translated into a single network address and its TCP/UDP ports. The NAPT is alternatively called **Port Address Translation, (PAT)** in the *Cisco* terminology. The PAT feature, extends NAT from "one-to-one" to "many-to-one" and can be used to translate several internal addresses into only one or a few external addresses, by associating the source port with each flow. PAT uses unique source port numbers on the private global IP address to distinguish between translations.

In the traditional NAT, sessions are uni-directional, outbound from the external network, in most cases. Sessions in the opposite direction may be allowed on an exceptional basis using static address maps for pres-selected hosts. Together the NAT and NAPT are referred as two variations of traditional NAT.

3.1.1.2 Translation Based Classification

Based upon the details of how NAT performs the translation process, NAT implementations can be classified into four classes: *Full cone NAT*, *Restricted cone NAT*, *Port restricted cone NAT* and *Symmetric NAT* (117).

- **Full cone NAT.** In this type of NAT, all requests from same internal IP address and port are mapped to same external IP address and port. Furthermore, any external host can send a packet to the internal host, by sending a packet to the mapped external address. This type of NAT is the simplest type of NAT and is rather easy for SIP to deal with - as we only need to determine what the external address and port number are for the client's private address and source port. Once this information is known, then this information can be placed into the SDP message.

- **Restricted cone NAT.** All requests from the same internal IP address and port, in this NAT, are mapped to the same external IP address and port. However, only the external host can send a packet to the internal host. Additionally, this external host can only send a packet to this internal host if the internal host has previously sent a packet to this host. Unfortunately, this means that this internal host is only available to a host that it has previously sent traffic to and this earlier traffic has to have been during the time that the address and port mapping is in the mapping table. This limitation is due to the fact that after some period of time the NAT will remove the mapping unless the internal host has sent additional traffic to the external host. The time before the NAT garbage collects "unused" mapping entries varies from NAT to NAT, thus an internal host and external host will experience unpredictable problems in communication as neither knows when the mapping entry for their communication will be removed.

- **Port restricted cone NAT.** A port restricted cone NAT is similar to a restricted cone NAT, but the restriction now includes the external host's port number. An external host can send a packet, with its source IP address and a particular source port, to the internal host, but only if the external host has previously sent a packet to this IP address and source port. If the internal host tries to send a packet from another source port, the packet will be blocked.

- **Symmetric NAT.** All requests from the same internal IP address and port to a specific destination IP address and port are mapped to the same external IP address and port. If the same host sends a packet with the same source address and port, but to a different destination, a different mapping is used. Furthermore, only the external host that receives a packet from this address and port combination can send a packet back to the internal host (with this specific address and port combination).

3.1.2 NAT Traversal: Problem and Solutions

In order to work properly, the NAT must have access to the protocol headers at Layers 3 and 4 (in case of a NAPT). Additionally, for every incoming packet it is required that the NAT already has a state listed in its table. Otherwise it will not be able to find the related internal host the packet belongs to. Since its appearance as a method, NAT seems to be solving quite well the issues with the IP addresses luck, increases the security and improves the administration. Proposed initially as temporary solution, today it becomes dominant in almost all networking scenarios.

Appears however, that not everything has been "transparent" with NATs. It fact it introduces on fly, many architectural implications and hence provoking issues and discussions, putting on question the advantage it brings.

The same characteristic that enhance privacy potentially makes debugging problem (including security violations) more difficult. If a host in stub domain (private network) is abusing the Internet in some way, such as trying to conduct attacks on another machine or sending large amount of spam, it makes more difficult to track the actual source of trouble because the IP address of the host address is hidden behind a NAT router. Therefore NAT inhibit implementation of security at the IP level.

Since there is still no suitable mapping created by outgoing packages, the external connection initiated between two endpoints, will be blocked by the NAT, causing failure of the entire communication passing through. In this sense, NAT breaks the end-to-end model and becomes critical infrastructural element. Therefore some prearrangement has to be made of incoming connection requests to traverse NAT devices.

The NATs casual use of private addresses makes collisions when companies using these addresses merge or want to directly interconnect using VPNs. The issues of media traversal is not straightforward either, and requires number of traversal methodologies.

According to RFC 3027 (81), the NAT traversal problem can be divided into three categories.

The first problem occurs if a protocol uses Realm Specific IP Addresses in its payload. That is, if an application layer protocol such as the Session Initiation Protocol (SIP) uses a transport address from the private realm within its payload signalizing where it expects a response. Since regular NATs do not operate above Layer 4, application layer protocols typically fail in such scenarios. A possible solution is the use of an Application Layer Gateway, ALG that extends the functionality of a NAT for specific protocols. However, an ALG only supports the application layer protocols that are specifically implemented, and may fail when encryption is used.

The second category are the Peer-to-Peer Applications. The traditional Internet consists of servers located in the public realm and clients that actively establish connections to these servers. This structure is well suited for NATs, because for every connection attempt coming from an internal client, the NAT can add a mapping to its table. But unlike client-server applications, a P2P connection can be initiated by any of the peers regardless of their location. However, if a peer in the private realm tries to act as a traditional server (e.g. listening for a connection on a socket), the NAT is unaware of incoming connections and drops all packets. A solution could be that the peer located in the private domain always establishes the connection. But what if two peers, both behind a NAT, want to establish a connection to each other? Even if the security policy would allow the connection, it cannot be established.

The third category is a combination of the first two. Bundled Session Applications, such as FTP or SIP/SDP, carry realm specific IP addresses in their payload in order to establish an additional session. The first session is usually referred to as the control session, while the newly created session is called the data session. The problem here is not only the realm specific IP addresses, but the fact that the data session is often established from the public Internet towards the private host, a direction the NAT does not permit (e.g. active FTP).(110) There are several techniques and solutions to solve these NAT traversal problems, mediate

SIP. Some of them include, Universal Plug-and-Play (UPnP), Simple Traversal of UDP through NAT (STUN), Traversal using Relay NAT (TURN) and Interactive Connectivity Establishment (ICE), in addition to manual configuration and application-layer gateway running on NAT. UPnP allows endpoints to punch "holes" on NAT for incoming requests and has difficulty with cascaded NAT or NAT in another domain. STUN relies on STUN servers to discover the existence and type of the NAT closest to the server, and depending on the type of NAT involved, traverses the NAT when possible. If not, TURN can be used to relay the connectivity between endpoints, i.e., the direct connectivity between endpoints is not achieved. ICE is an integrated framework of these approaches and attempts to traverse NAT from the most optimistic to pessimistic way. In continuation, follows a full observation of the above techniques and introduction of some additional ones.

3.1.2.1 Session Traversal Utilities for NAT (STUN)

Session Traversal Utilities for NAT, STUN (116) is used as NAT traversal method for interactive IP communications. It provides a mechanism for the client to discover whether it is behind an NAT, what the specific the type of NAT is, and what mapping the NAT has allocated for this client's private IP address and port (i.e., what public IP address and port corresponds to this private IP address and port).

STUN is a client-server protocol. If STUN-enabled SIP client sends a request to bind its private IP address and public IP address to an STUN server, the NAT will modify the source transport address and port number of this packet when the binding request message passes through the NAT. After the STUN server receives this packet, it sends a binding response back to the STUN client containing the client's mapped IP address and port on the public side of the NAT. When the packet passes back through the NAT, the NAT will modify the destination address to be the client's private IP address and the client will Receive the STUN response. Now the client knows its external public address and port combination (at least in terms of what mapping the NAT did when sending a packet to the STUN server from this client's private IP address and source port). The mapped public IP address and port provided by the STUN server can be used as the value in the "Contact" field in a SIP call establishment message, thus the Contact field contains a valid globally routable IP address and port.

Depending upon the type of NAT, different address and port mapping schemes will have been used. STUN works primarily with three types of NAT: full cone NAT, restricted cone NAT, and port restricted cone NAT. An obvious drawback of STUN is that it does not work with a symmetric NAT, as this type of NAT will create a mapping based on internal IP address and port number as well as the destination IP address and port number. When the destination IP address of the SIP proxy is different from the address of the STUN server, then the NAT will create two different mappings using different ports for traffic to the SIP proxy and STUN server, thus the mapping which STUN learned and which will be used during SIP call establishment messages is incorrect. As a result the SIP signaling will not be correct and the session will not be setup properly when a SIP client is behind a symmetric NAT.

STUN provides one solution for an SIP application to traverse the NAT, as it allocates a public IP address and port for the client and allows the client to receive packets from a peer with this transport address. However, a STUN server does not permit the client to communicate with all peers with the same transport address (public IP address and port). This lead to the development of another solution that could address the problem, we describe this solution in the next subsection.

3.1.2.2 Traversal Using Relay NAT (TURN)

Traversal Using Relay NAT (TURN)(118), an extension to the STUN protocol, is designed to solve the symmetric NAT traversal problem. Because the problem caused by a symmetric NAT was that the external (public) IP address and port for the SIP client outside the NAT would be different if the packer were to be sent to another global IP address and port, the solution is for the TURN server to relay packets to and from other peers. In this way the mapping that the TURN client learns is correct and the packets that the SIP client sends will be relayed by this TURN server.

If TURN-enabled SIP client sends an exploratory request to the TURN server, a binding response containing the client's mapped IP address and port on the public side of NAT is sent back. This mapped IP address and port are used in both SIP call establishment messages and media streams. The TURN server relays packets to the client when a peer sends data packets to the mapped address. Although a TURN server enables this client to communicate with other peers, it comes at a high cost to the provider of the TURN server as this server needs a high bandwidth connection to the Internet, since the amount of traffic across this connection is twice the volume of relay traffic - as all the traffic has to go both to the TURN server and from the TURN server to the relay target. Moreover, like STUN, TURN requires SIP clients to be upgraded to support its mechanism.

3.1.2.3 Universal Plug and Play

Using an UPnP controllable Internet Gateway Device (IGD) (129) is probably one of the most reliable methods for UDP and TCP. Unfortunately, UPnP has such strong security issues that it is often disabled and cannot be used with many NATs. In (50) authors' test results show that UPnP was only enabled in 36.42% of all NAT devices. The advantage of UPnP over other techniques is the easy allocation of new mappings. An UPnP mapping does not need to be aware of the source of the connection, it is enough to specify which external port is forwarded to which internal port.

UPnP is a network protocol for automatic discovery and configuration when a certain device (i.e., an UPnP client) is on-line. Therefore, IP information mappings in both the UA and the NAT can be automatically established by the UPnP protocol. Then all SIP/RTP packets traverse over the NAT.

Interactive Connectivity Establishment (ICE)

ICE (119) is a form of peer-to-peer NAT traversal that works as an extension to SIP. ICE provides a unifying framework for using STUN and TURN around it. ICE first tries STUN to get through NAT (because STUN method doesn't need middle nodes to relay media data). When a host is behind symmetric NAT, another another host is behind the other NAT, traffic data transmitted between two hosts has to be relayed by middle nodes which only use TURN solution to get through. All the ICE's connecting check processes are set up from the NAT inside to avoid the anonymous connecting request to be blocked and dropped by NAT. The detailed operation of ICE can be broken into six steps: (1) gathering, (2) prioritizing, (3) encoding, (4) offering and answering, (5) checking, and (6) completing.

Gathering: Before making a call, the ICE client begins by gathering all possible local IP addresses and ports from interfaces on the host. These potential IP addresses and ports are called host candidates. Then, the client contacts the STUN server from each candidate to learn the pair of public IP address and port which are allocated by NAT for this candidate. These public IP address and ports are called *server-reflexive candidates*. Finally, the client contacts the TURN server and obtains relayed candidates.

Prioritizing: Once the client has gathered its server-reflexive candidates and relayed candidates, it assigns a priority value for each of them.

Encoding: After the gathering and prioritizing processes, the client constructs its SIP INVITIE request to set up the call. ICE adds host candidates, server-reflexive candidates and relayed candidates as candidate attributes in SDP attributes for the SIP Request message.

Offering and answering: The SIP network send the modified request to the called terminal. The called terminal generates a provisional SIP response which contains the candidate information of the called terminal.

Checking: Through the above processes, the caller and called terminal have exchanged SDP messages. The caller and called terminal pair each of its candidates with a candidate from its peer. ICE uses a STUN transaction to check if a candidate pair works or not. This check is conducted in priority order and the highest-priority pair will be used for the subsequent traffic.

Completing: The caller generates the final check to its peer to confirm the highest-priority candidate pair as the one which will be used later. Finally, the media traffic begins to flow.

Although ICE combines the benefits of STUN and TURN without their disadvantages, it is still not a flawless solution and the drawback is both obvious and intolerable for the users. It inevitably increases call-setup delays as all of the gathering and checking takes place before the called terminal even receives the SIP INVITE. It also has disadvantages for the NAT, in that each of the candidates leads to the allocation of a server-reflexive candidate, thus taking up public IP address and port combinations that can not be used by another client inside the private network. While this might not be a problem for a single user at home, it can be a problem for a mobile operator who is using a NAT between their mobile packet data network and the public Internet.

3.1.2.4 Virtual Intranet Platform (VIP)

VIP (136), is a P2P Communication Platform solution for NAT traversal, which use the public DHT service - OpenDHT as the distributed address/port information rendezvous. Without changing the configuration of the NAT, all the network and distributed application service behind the NAT can make use of the VIP to communicate with the corresponding peer services outside the NAT. The performance of the bandwidth, data lost and delay problems are much better than the existing traditional client-server framework platforms, more general than specific P2P applications. The P2P Communication Platform for NAT Traversal-VIP, is robust and scalable because there are no single failure points in the platform, the structure is distributed, and majority of the traffic data between two hosts behind the NAT can be transfer directly without relaying.

3.1.2.5 Advanced NAT Traversal Service (ANTS)

ANTS (50), is a draft version of framework improving the communication of existing and future applications across NAT devices. The core idea of ANTS is to use previously acquired knowledge about NAT behavior and services for setting up new connections. It purposes signaling protocol for the coordination of distributed instances. In a comparison to ICE, it is claimed ANTS to be not only more flexible, but also faster due to the decoupled connectivity checks.

3.1.2.6 Multiple Subscriber Videoconferencing System

This patent application (87), represent a system, method and device for video conference. It includes installing videoconferencing switch at an access point to an IP network and registering subscribers for the videoconferencing services, each one having plurality of endpoints. Furthermore, this method includes receiving subscriber-specific settings to be applied to multiple videoconferencing calls from the plurality of endpoints associated with each subscriber. The method further includes storing the subscriber's specific settings at a location accessible to the switch, and configuring the switch to connect calls from the plurality of endpoints at each subscriber based on the corresponding subscriber-specific settings.

In this invention, the security settings like firewall and NAT are on a pre-subscriber basis, i.e., are configured within the switch to provide network address translation services from a group consisting of SIP network address translation and H.323 network address translation. H.232/SIP applications parse control data to dynamically open and close port for control of traffic. The information obtained from parsing is sent to network data plane hardware. Configuring firewall includes adding address information into gatekeeper for the zone, setting ports or channels that are statically open and setting security log in.

For the H232 NAT module, the NAT module is configured to parse packet headers and payload of Q.931/H.245 control data streams during call set-up. For outgoing data, NAT module is further configured to substitute non-routable endpoint source IP addresses and port numbers with its own globally unique H.323 proxy IP address and port numbers. For incoming data, the NAT substitutes non-routable, or internal endpoint destination IP addresses and port numbers by using stored IP address/port number mapping information.

3.1.2.7 Traversal of non-Protocol Aware Firewalls and NATs

Besides the described methods, (64) puts up another scheme that is called "Traversal of non-Protocol Aware Firewalls and NATs". This solution uses a combination of techniques including tunnels and probe packets to traverse NAT/Firewall. The system does not require firewalls and NATs to be upgraded. Moreover, it has a good security by using tunnel method.

3.1.2.8 A New Tunnel Scheme for Multimedia Communications Traversing NAT/Firewall in NGN

Test results in (141), proves that their presented scheme can implement the main function of traversing NATs and firewalls. However compared with solutions without tunnel, this method makes some delay to call set-up time due to the deployment of Call Client Agent (CCA). This Scheme is based on MASMCT (75), two-layer multi-agent architecture comprised of client and server agents.

3.1.2.9 MASMCT

"MASMCT" is used by service providers, to resolve the issue when they want to build a large-scale Next Generation Network (NGN) with many NATS and firewalls to enterprises and personal subscribers. It is easy to set up because it doesn't need to upgrade the existing device like firewall, NAT or endpoint device of subscriber. Compared with many solutions, it has a good security by using TLS and tunnel method. It's also easy to support those protocols that use TCP connection, such as H.323.

3.1.2.10 Nat friendly SIP

"Nat friendly SIP" (120), states that one way to handle SIP, given the existence of NATs is to embed a SIP ALG within enterprise NATs, thus stopping SIP-unaware commercial NAT products. The draft, proposes solutions for getting SIP through enterprise and residential NATs that does not require changes to these devices or to their configurations.

3.1.2.11 Midcom-unaware NAT/Firewall Traversal

In the draft "Midcom-unaware NAT/Firewall Traversal" (123), a pre-Midcom solution framework is developed. Two key components of the proposed solution are: (1) a Signaling Proxy server on the signaling path, and (2) a Media Proxy server on the media path. In the context of SIP and RTP, the signaling path component will be a SIP Proxy server called Back-to-Back-User-Agent (BBUA) and the media path component is an RTP Proxy. The signaling and media proxies interact using some control protocol, which is transparent to the end users. The signaling path is always kept open through the NAT using a keep-alive mechanism. The public IP address/port allocated by the NAT is returned to the Client in the Contact header of 200 OK sent by the Proxy server. This allows the UA to perform NAT/FW failure detection by - (a) not receiving 200 OK over a prolonged period of time, and (b) detecting that the NAT IP address has changed in a received 200 OK.

3.1.2.12 Application Layer Gateway (ALG)

Application Layer Gateway (ALG) and Session Border Con- troller (SBC) are server-based solutions widely used in the SIP-based VoIP environments. SIP-

ALG typically collocates with the NAT to create the private-to-public IP information mappings and uses these mappings to translate the SIP messages

3.1.2.13 Middle box Communications (MIDCOM)

The MIDCOM protocol allows the middle-box to perform its operation with the aid of "MIDCOM agents", without resorting to embedding application intelligence. The principal motivation behind he architecture of this protocol is to enable complex applications through middle-boxes seamlessly using a trusted third party, i.e., a MIDCOM agent. It is important to note that an agent and a middle-box can be on the same physical device. In such a case, they may communicate using a MIDCOM protocol message formats, but using a non-IP based transport such as IPC messaging (or) they may communicate using a well-defined API/DLL (or) the application intelligence is fully embedded into the middle-box service (as it is done today in many stateful inspection firewall devices and NAT devices). The MIDCOM protocol consist of a session setup phase, run-time session phase and a session termination phase.

Application Layer Gateway, ALG (114) and Middle box Communications, MIDCOM (127) are all the classic NAT traversal solution. These methods work by changing the NAT configuration more or less to get through NAT. Generally speaking, ordinary users behind the NAT have no right to change NAT configuration. So these NAT Traversal solutions are not suitable for the general P2P communication applications.

There are circulating numerous techniques and architectural solution for the problem of NAT traversal in multimedia scenarios. Finally (126) presents a very good summary of the most popular NAT traversal techniques.

Even with these NAT traversal techniques, let incoming connection requests traverse NAT and reach the endpoints behind NAT is still a nontrivial job, especially for symmetric NAT that allocate different port numbers for different internal and external endpoint pairs (77).

NAT traversal techniques are typically required for client-to-client networking applications on the Internet involving hosts connected in private networks, especially in peer-to-peer and Voice over Internet Protocol (VoIP) deployments (71).

As stated previously, NAT is particularly hard problem for P2P applications, which rely on direct communication between peers. Such a P2P application is **BitTorrent**, for which the Internet measurements have shown that most of its users are behind NAT(79). This is particularly a problem with traditional NAT and may be less of an issue with bi-directional NAT, where sessions are permitted in both directions. In (102) are conducted mathematically tractable models to study the performance of BitTorrent-like P2P systems with the presence of homogeneous and heterogeneous NAT peers. The authors suggest interesting strategy to improve the performance of NAT peers and that of the entire system.

In (69) the author address the NAT traversal problem in P2P application offering so-called **NatTrav** approach, based on intermediate connection broker, used to register the communicating peers, provide current network address for the recipient peer and facilitate NAT traversal if the recipient is NATed, through their "NAT Traversal Protocol for cone-type mappings".

Some excellent P2P systems such as **Skype** (53), have solved NAT traversal problem. But most solutions are just focus on the specific applications and can't be easily re-used by other application. For example, the NAT Traversal solution of Skype can't be used by other P2P applications. The ideal goal is to find out a general solution of NAT traversal, which can easily be co-used by many P2P applications.

A possible work-around for this type of problem with traditional-NAT is for private hosts to maintain an outbound connection with a server acting as their representative to the globally routed Internet(81).

UDP hole punching is a commonly used technique employed in NAT applications for maintaining User Datagram Protocol (UDP) packet streams that traverse the NAT. UDP hole punching enables two clients to set up a direct peer-to-peer UDP session with the help of a well-known rendezvous server, even if the clients are both behind NATs. This technique was mentioned in (81), documented more thoroughly elsewhere on the Web and used in experimental Internet protocols (82).

As existing protocol designed explicitly to work with NAT enroute, that also use UDP hole punching, such that does not require an ALG for the games to work transparently through traditional NAT devices is, **Activision Games** (81).

3.2 An Advanced Multimedia Communications System

The analysis of the problem are observed in a sub-scenario of an already existing project *Vision*[1] (37). The goal of the project *Vision* is to develop new environment for deployment of several purpose technologies like technology for reality catch, technologies for creation of systems for video acquisition, advanced technologies for analysis and video processing, advanced technologies for analysis and audio processing, advanced communication technology, and technologies for presentation of reality. In the frames of this project, it was set a scenario that served as a test bed for the issues to examine. The obtained results were based

[1] *The project "Vision" takes part of the DIT Department, ETSIT, UPM*

on simulations.

3.2.1 Description of the *Vision* scenario

The problem tackled in this chapter, is established in the environment of one out of the three kinds of scenarios described in the Vision project. The scenarios are divided in three categories - Scenario *Lite*, *Entertainment* and *Corporate*. All of them represent distant communication in a form of video conference among various participants used for different causes, thus the three different names of the scenarios.

The *Lite* scenario describes 3D real time communication between two distant participants, with the possibility of multimedia material exchange. *Entertainment* simulates various distant visual 3D rooms, each containing participants practicing Japanese martial art skill *"Tai Chi"*, and one room with professor inside. All participants communicate simultaneously sending and receiving information through each other. In the last *Corporate* scenario, it's presented the most advanced communication, a business meeting among various participants, all placed in their headquarter offices in different cities. The form of communication appears similar to the previous scenario, with exception that all participants have equal role. They send and receive information as signals or media, with possibility to do other things meanwhile, like showing presentations, recording the meeting and so on.

We were concentrated on the last scenario, as most complex containing the general architecture for all three scenarios. I won't enter into any further details, rather I will focus on description of our own scenario including all the properties of the specified conversation.

Briefly, the basic event flow in the scenarios is the fallowing. The organizer of the meeting communicates with the application *"Advanced Manager of Virtual Meetings"* announcing the type of the meeting (lite, entertainment, corporate), specifying all the details of the place, time, people and place of the meeting, confirming their availability. The messages are sent through IMS network using the standard functions for localization, presence and messaging in IMS. After the configuration of the virtual scenery, it is created the absolute position for each participant in the virtual room and the view list of the rest of the participants. Notifications are sent to all participants while physical resources are reserved. Finally the systems takes care to send to all of them, the configuration file of the meeting.

The requirements of the application Vision will be approached closely from a point of view of the communication flows inter-exchanged among the participants. The third scenario will be considered that involves simultaneous communication among three different participants, placed in three distant rooms.

Requirements in the interface of the user, there should be considered are the fallowing:

- Photography compare: Use of a *Photolog* application
- System of recordings and tags: Small menu system to offer the user different options like (REC, PAUSE, STOP, TAG)

For the rendering part, there exist also some specific features. One of the characteristics that underpin this first scenario is the visualization out of the depth maps, which implies the definition of sending this information from the reconstruction module to module display. Two plug-ins are considered to receive the flows. First, the renderSink that receives the flows for 3D reconstruction (VGA and 10FPS) and the plug-in "HD + DepthSending" receives the views Left, Right, and Depth required for 3D video conferencing. Below are defined the incoming and outgoing flows of each plug-in:

- Plugin RenderSink. Receives 18 streams VGA of 10FPs of the plug-in SelectView. It sends four VGA flows to eStudio using the communication protocol developed by Brainstorm. During the events, Gstreamer indicates preference order of the views
- Plugin "HD + DepthSending". Receives from SelectView the fallowing three flows and sends them to eStudio using the previous protocol
- Left. Corrected image from the left camera at 1280x720 and 23FPS
- Right. Corrected image form the right camera at 23FPS and 1280x720
- Depth. Depth Map recorded at the left camera "Left" at resolution 744x416 and 23FPS

3.2.2 Proposed Solution for the Scenario

For the purpose of our goal [1], we considered the flows described in the previous section into the fallowing multimedia scenario. All flows are captured in six different machines on the side of each user, independently. In order to communicate the rest of the participants, each machine has assigned IP address. All of the

[1] *The experimental work done in this section is a cooperative work established for the purpose of one subject in the Master*

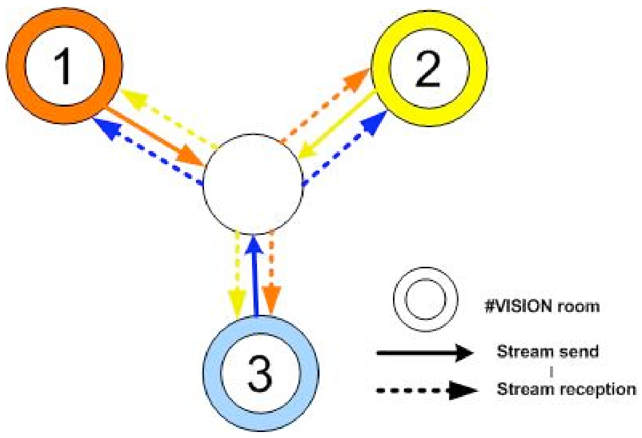


Figure 3.1: Participants Communication Interconnection

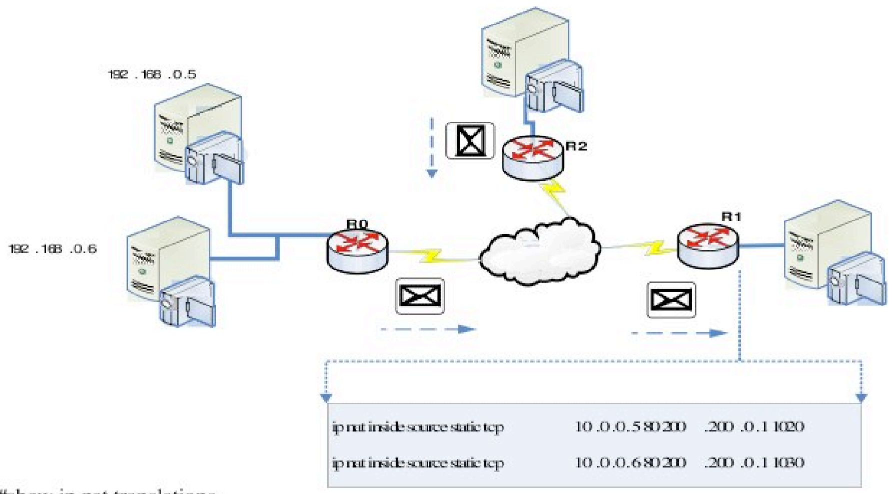
three participants, has a separate lines to send and receive information to/from each of the rest, as shown in the Figure 3.1.

Videoconferencing over IP networks faces many problems, including security, bandwidth utilization, quality of service and deployment and management. H.323 and SIP are approved standards to enable compatibility in multimedia communication, particularly teleconference transmissions over IP networks.

NAT rewrites IP headers as packet passes through the device. The NAT device maintains a table of mapping between an IP addresses and port numbers. The problem with sending SIP traffic in the case of our scenario, through a NAT, is that this protocol makes heavy use of embedded IP addresses, while normal data traffic contain IP address in the header of each packet. While configuring a NAT to rewrite packet headers to change addresses is relatively simple, the difficulty comes in configuring a NAT to translate addresses embedded in SIP traffic, because the location of these address in these data stream is difficult to calculate.

For our case, having many IP address on each side, make the communication more complex to handle, especially in the interconnection part of the users. As noted earlier, all communication lines are bridged through switch and router to Internet. Initially we considered implementing the NAT technique in the routing algorithms on each side, only for IPv4 addressing issues, omitting the security as an issue. Despite that, we wanted to connect all devices in the multimedia communication (six in this case), to a common router, such that the three routers will be the ones sharing the intercommunication. This means that we want to translate the six IP address in one, before announcing it furthermore, so that each recipient (participant in the video conference) would only see the source of the packets they receive in a form of a unique IP address. This is supposed to optimise the entire topology. Immediate consequent issue appears to be the transfer of all data flows simultaneously with minimum possible lost.

The initial set up of the flow measurements as well as the entire picture of the scenario, is presented in Figure 3.1. It presents the network topology, and the type of flows in each communication line, together with the required bandwidth. The flows can be signals of media data packets, clients can send and receive data simultaneously. Note, the table on the right side summaries the quantity of flows, their source and bandwidth.



```
ip nat inside source static tcp        10 .0.0.5 80 200 .200 .0.1 1020
ip nat inside source static tcp        10 .0.0.6 80 200 .200 .0.1 1030
```

```
R0#show ip nat translations
Pro  Inside global      Inside local          Outside local        Outside global
tcp 171.68.1.1:1020     192.168.0.5:1020      200.200.0.1:1020     200.200.0.1:1020
tcp 171.68.1.1:1020     192.168.0.5:80              ---                  ---
tcp 171.68.1.1:1030     192.168.0.6:80              ---                  ---
R1#show ip nat translations
Pro  Inside global      Inside local          Outside local        Outside global
tcp 200.200.0.1:1040    10.0.0.5:80                 ---                  ---
tcp 200.200.0.1:1020    10.0.0.5:80           171.68.1.1:1020      171.68.1.1:1020
tcp 200.200.0.1:1030    10.0.0.6:80                 ---                  ---
```

Figure 3.2: PAT Translation

3.3 Simulations

We wanted to examine intercommunication of the participants, i.e., sending and receiving of flows to the extent that we will be fallowing the exchanging packets' behavior and the undergoing NAT translation. For that purpose, we used two simulation environments, Cisco's packet Tracer version 5.2 and the open source GNS3. The choice of these tools, was due to the fallowing guidelines:

GNS3 is a free source simulator, that permits emulation of a quite real network environment, with routers and switches. It enables routing and packets sending tests. CNS3 appears to be powerful tool for debugging tests, by allowing operational functions as of a real router. However, not possessing network cards and ability to simulate computers, omitted us of using its entire potential, therefore Packet Tracer helped as another alternative for that purpose.

Packet Tracer, on the other hand, is a *Cisco* developed environment, enables essential tools for network simulation like routers, switches including servers. It enables NAT tests as we required, ability to specify ports and verify Port Address Translation (PAT) in a real environment. Moreover, with PT we've been able to simulate computers acting as a servers, in order to test the PAT over the packets of video streams.

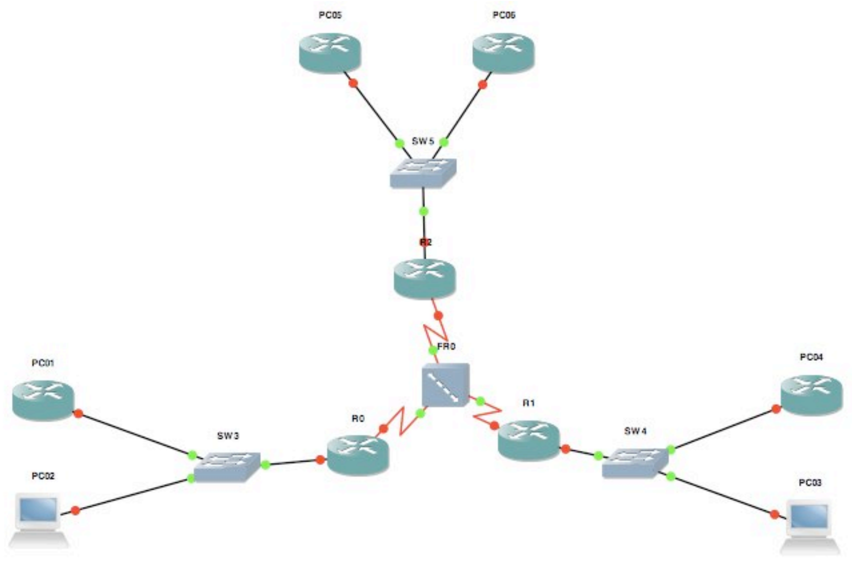


Figure 3.3: GNS3 Simulation Topology

Figure 3.2 in continuation depicts shorten version of our topology (including the three clients, one containing two IPs for the video streams and the rest only one IP), in which one packet is being generated from the server behind the router R0, towards the router R1. It demonstrates correct matching of the port number in the inside local and global IPs with the outside local and global IPs. This ensures correct packet delivery to the desired recipients onto the desired ports.

We established the first simulation model in GNS3. It initially consisted of three participants/endpoints, each one sending two packet flows to the rest, hence having two IP addresses on each side. Fallowing step by step modifications, we built the entire set up and attached two devices so we could latter add up the rest of the topology.

Figure 3.3 represents our simulation topology. Suppose we have three distant participants behind routers interconnected in a star through frame relay switch. Since there are six (or in our case two) IP lines on each side, we bridge them to the routers through Ethernet switch. The choice of routers fell on *Cisco 7200 Series* routers, as widely developed highly potential one, reaching bandwidth of up to 2 Mbps.

Next we set up interfaces between the devices, configured the routers R0, R1 and R2, assigned them IP addresses, tested their mutual communication and finally built up the access tables in each of the routers.

Proceeding with forced "ping" in order to simulate sending packets form one router to another, we aimed at sending form router R0 to R1 specifying as a source address 192.168.0.1 of R0 and 200.200.0.1 as target address of the router R1, both private. This type of simulation allowed us to emulate the end parts or the communicating machines, i.e. to be able to represent the flows as interchanged data packets between the routers. After consulting the NAT table we confirmed that NAT translation was done correctly.

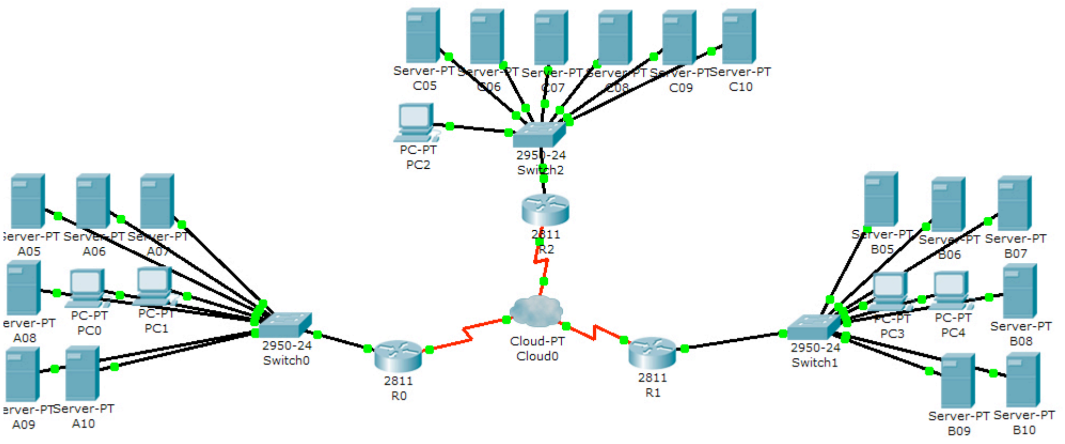


Figure 3.4: Packet Tracer Simulation Topology

Furthermore, a loop-back was created and attached to router R0, as virtual interface to sends packets instead of a machine. We give permission to the routers R1 and R2 in the access list of the router R0. By doing this, we enabled routing between the three routers, and hence allowing access to proper exchange of packets preventing NAT as obstacle on their way.

Figure 3.4 depicts the same topology from GNS3, just this time in Packer Tracer. As mentioned previously, this simulation environment enables us proceed accurate NAT and PAT translation, modeling the participating machines as servers and mediate computers attached to the switches, to invoke desired servers on the other side by sending requests and packets. This means, it is possible to specify the port of the machine we want to communicate/send packets to and verify that PAT translation is done correctly.

Analogically as in GNS3, we've done all the setups and tests regarding sending messages and building access tables. Without going into further details of the simulation, results confirmed full feasibility of the network topology. Outcome from the NAT tables, states not only accurate NAT by also correct PAT translation. The messages were delivered successfully, without any significant delay.

Chapter 4

Cloud Computing Paradigm

4.1 Introduction to Cloud Computing

Cloud computing is a cutting edge paradigm in the business world and becomes very popular topic discussed in various scientific papers, conferences and on the Internet. In the same time correlated with many other technologies, it attracts the attention of the research world that focuses in relying on the cloud structure when developing various topologies and applicable solutions.

It represents a complex utility that includes applications as a services distributed over the Internet together with the hardware and system software in the data centers that provide those services. The cloud represents the data center hardware and software managed by the provider who takes care of all the maintenance and control details, omitting the infrastructural organization form the user.

The term "cloud" is used as a metaphor for the Internet, and was inspired by the cloud symbol used in the past to represent the telephone network, and later to depict the Internet in computer network diagrams as an abstraction of the underlying infrastructure it represents.

The huge popularity for this novel area appears as a result of the big marketing champagnes proceed by the initial and leading providers of cloud services like Amazon, Google, IBM, Yahoo, Microsoft, etc. However the rhythm of its rapid growth can be explained with the significant innovations in virtualization and distributed computing, especially inspired by the improved access to high-speed Internet and the need for reliable services delivered through data centres and built on servers.

If 50 years ago we had to adopt the time-sharing servers due to limited computing resources, today the Cloud computing comes into fashion with the need to build complex IT infrastructures. Users have to manage various software installations, configurations and updates. Computing resources and other hardware are prone to be outdated very soon. Therefore outsourcing computing platforms is a smart solution for users to handle complex IT infrastructures.

4.2 Definitions and Classification

Currently, the cloud computing is still in an evolving phase, thus no widely accepted general definition is provided yet. It is generally accepted that cloud computing refers to a **new IT paradigm for users**. However, in the literature and on Internet are circulating various definitions. Following are some of them:

"A computing Cloud is a set of network enabled services, providing scalable, QoS guaranteed, normally personalized, inexpensive computing infrastructures on demand, which could be accessed in a simple and pervasive way." (131)

"People are coming to grips with Virtualization and how it reshapes IT, creates service and software based models, and in many ways changes a lot of the physical layer we are used to. Clouds will be the next transformation over the next several years, building off of the software models that virtualization enabled" - Douglas Gourlay (12).

"The "Cloud" concept is finally wrapping peoples' minds around what is possible when you leverage web-scale infrastructure (application and physical) in an on-demand way. "Managed Services", "ASP", "Grid Computing", "Software as a Service", "Platform as a Service", "Anything as a Service"... all terms that couldn't get it done. Call it a "Cloud" and everyone goes bonkers. Go figure" - Damon Edwards (12).

"Cloud computing is ... the user-friendly version of grid computing" - Trevor Doerksen (12).

"A computing capability that provides an abstraction between the computing resource and its underlying technical architecture (e.g., servers, storage, networks), enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction." (11)

A cloud can be private or public. A **public cloud** sells services to the public over the Internet for only as much capacity as is needed, and bring more resources online as soon as required. This "pay-as-you-go" model resembles the way electricity, fuel and water are consumed, therefore is referred to as *utility computing* (Currently, Amazon Web Services is the largest public cloud provider.)

A **private cloud** is a proprietary network or a data centre that supplies hosted services to a limited number of people, when they are large enough to benefit from the advantages of cloud computing.

Thus, cloud computing is the sum of SaaS and utility computing. Increased benefits can be achieved by combining private with public clouds in a *hybrid cloud computing*. Private or public, the goal and the key necessary enabler of cloud computing is to provide easy access to computing resources and IT services and the construction and operation of extremely large-scale, commodity-computer data centers at low-cost locations.

When a service provider uses public cloud resources to create their private cloud, the result is called a **virtual private cloud**. Virtualization enables automatic allocation and management. According to (51), different utility computing offerings will be distinguished based on the cloud system software's level of abstraction and the level of management of the resources.

Cloud Computing technology offers different service models or various classes of utility computing as a base for successful end user applications. Those models are group in three kinds of cloud services (Figure 4.1) that currently exist:

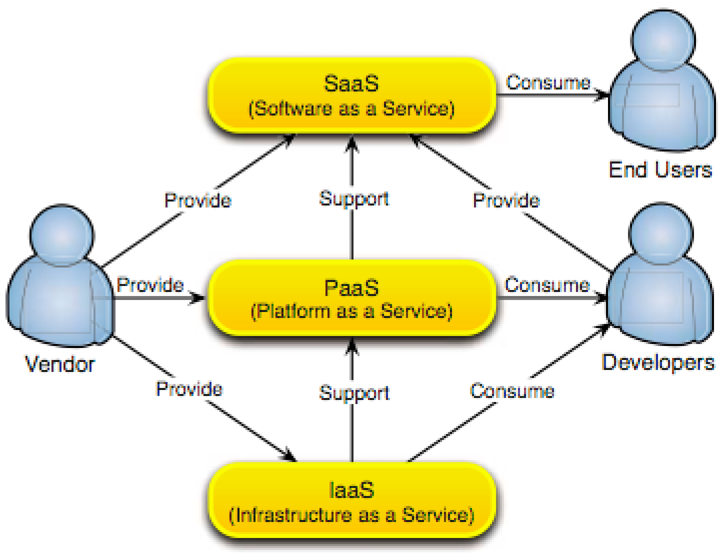


Figure 4.1: Computing paradigm shift in six phases

- **Infrastructure as a Service (IaaS)**, provides low-level services like virtual server instances with unique IP addresses and blocks of storage on demand which can be booted with a user-defined hard disk image, for example **Amazon Web Services and EC2**. An EC2 instance looks much like physical hardware, and users can control nearly the entire software stack, from the kernel upward. Virtual hard disks that can be accessed from different virtual machines are another example of infrastructure as a service. Customers use the provider's application program interface (API) to start, stop, access and configure their virtual servers and storage.

- **Platform as a Service (PaaS)**, in the cloud is defined as a set of software and product development tools hosted on the provider's infrastructure. Developers create applications on the provider's platform over the Internet. PaaS providers may use APIs, website portals or gateway software installed on the customer's computer. **Force.com**, (an outgrowth of Salesforce.com) and **Google AppEngine** are domain-specific platforms, examples of PaaS, which are targeted exclusively at traditional Web applications. Microsoft Azure is intermediate between application frameworks like AppEngine and hardware virtual machines like EC2. Developers need to know that currently, there are not standards for interoperability or data portability in the cloud. As a result, some providers will not allow software created by their customers to be moved off the provider's platform.

- **Software as a Service (SaaS)**. Here the vendor supplies the hardware infrastructure, the software product and interacts with the user through a front-end portal. SaaS is a very broad market. Services can be anything from Web-based email to inventory control and database processing. Because the service provider hosts both the application and the data, the end user is free to use the service from anywhere. Examples are Web-based office applications like Google Docs or Calendar, but also the upcoming gaming service by Onlive. SaaS is usually build based on own or foreign IaaS and/or PaaS.

Figure 4.2 depicts the computing paradigm shift of the last half century. More specifically, the figure identifies six distinct phases. In the first phase, people used terminals to connect to powerful mainframes shared by many users. Back then, terminals were basically little more than keyboards and monitors.

In Phase 2, stand-alone personal computers (PCs) became powerful enough to satisfy users daily work - you didn't have to share a mainframe with anyone else. Third phase presents the computer networks that allowed multiple computers to connect to each other. You could work on a PC and connect to other computers through local networks to share resources.

Phase 4 saw the advent of local networks that could connect to other local networks to establish a more global network-users could now connect to the Internet to utilize remote applications and resources. Phase 5 brought us the concept of an electronic grid to facilitate shared computing power and storage resources (distributed computing). People used PCs to access a grid of computers in a transparent manner.

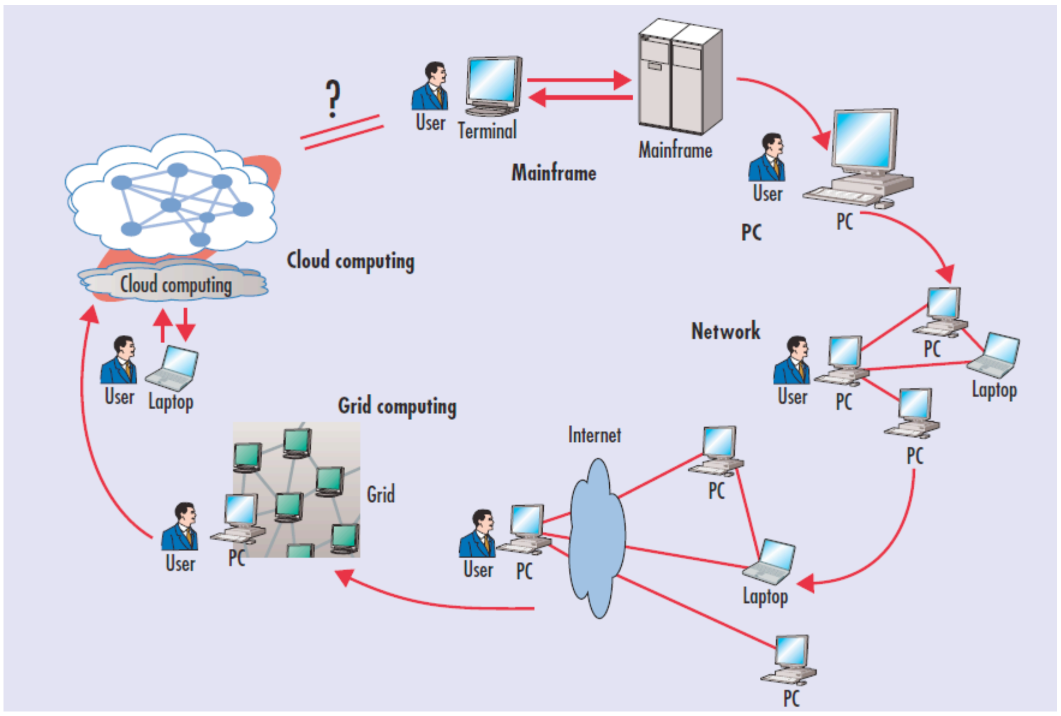


Figure 4.2: Computing paradigm shift in six phases

Now, in Phase 6, cloud computing lets us exploit all available resources on the Internet in a scalable and simple way (130).

What makes the cloud computing new and different from the traditional hosting seen from a hardware provisioning and pricing point of view above all, is the appearance of infinite computing resources available **on-demand** for which you can pay in a short-term basis for what you use (for example, processors by the hour and storage by the day) and release them as needed which leads potentially to increased cost savings. Moreover it has a characteristic of **elasticity** or users can "rent" (add or remove) resources (services, space) at any time they want and be charged precisely, according to time measured in minutes rather than weeks thereby allowing companies to start small and increase hardware resources only when there is an increase in their needs.

Usage-based pricing is not renting. Renting a resource involves paying a negotiated cost to have the resource over some time period, whether or not you use the resource. **Pay-as-you-go** involves metering usage and charging based on actual use, independently of the time period over which the usage occurs. Real world estimates of average server utilization in data centers range from 5% to 20%. Moreover the cloud computing offers self-service, broad network access, resource pooling, rapid elasticity, and measured service.

The achievement in cloud computing is that it could offer services below the costs of a medium-sized data center and yet still make a good profit, i.e., using 1,000 servers for one hour costs no more than using one server for 1,000 hours. Cloud computing is often described as "converting capital expenses to operating expenses" (CapEx to OpEx), but the phrase "pay as you go" more directly captures the economic benefit to the buyer.

4.3 Market-Oriented Cloud Architecture

The majority of cloud computing infrastructure, consists of reliable services delivered through data centers and built on servers. Clouds often appear as single points of access for all consumers' computing needs. As consumers rely on cloud providers to supply more of their computing needs, they will require from their commercial offerings to meet quality of service (QoS) in order to meet their objectives and sustain their operations, negotiating in the same time specific SLAs.

The very first issue is the definition of SLA specifications in such a way that has an appropriate level of granularity, namely the tradeoffs between expressiveness and complicatedness, so that they can cover most of the consumer expectations and is relatively simple to be weighted, verified, evaluated, and enforced by the resource allocation mechanism on the cloud. In addition, different cloud offerings (IaaS, PaaS, SaaS) will need to define different SLA meta-specifications.

To achieve this, market-oriented resource management is necessary to regulate the supply and demand of cloud resources to achieve market equilibrium (where supply = demand), providing feedback in terms of economic incentives for both cloud consumers and providers, and promoting QoS-based resource allocation mechanisms that differentiate service requests based on their utility. In addition, clients can benefit from the "potential" cost reduction of providers, which could lead to a more competitive market and thus lower prices (55).

4.3.1 Emerging Cloud Platforms

Industry analysts have made bullish projections on how cloud computing will transform the entire computing industry. Cloud computing is expected to be a $160-billion addressable market opportunity, including $95-billion in business and productivity applications, and another $65- billion in online advertising. Another research studies identify cloud computing as one of the prominent technology trends.

As the computing industry shifts toward providing Platform as a Service (PaaS) and Software as a Service (SaaS) for consumers and enterprises to access on demand regardless of time and location, there will be an increase in the number of cloud platforms available. Recently, several academic and industrial organizations have started investigating and developing technologies and infrastructure for cloud computing as for example OpenNebula (29). Following are described some of the representative cloud platforms with industrial linkages.

Amazon Elastic Compute Cloud (EC2) (3) provides a virtual computing environment that enables a user to run Linux-based applications. The user can either create a new Amazon Machine Image (AMI) containing the applications, libraries, data and associated configuration settings, or select from a library of globally available AMIs. The user then needs to upload the created or selected AMIs to Amazon Simple Storage Service (S3), before he can start, stop, and monitor instances of the uploaded AMIs. Amazon EC2 charges the user for the time when the instance is alive, while Amazon S3 (5) charges for any data transfer (both upload and download).

Google App Engine (18) allows a user to run web applications written using the Python programming language. Other than supporting the Python standard library, Google App Engine also supports Application Programming Interfaces (APIs) for the data-store, Google Accounts, URL fetch, image manipulation, and email services. Google App Engine also provides a web-based Administration Console for the user to easily manage his running web applications. Currently, Google App Engine is free to use with up to 500MB of storage and about 5 million page views per month.

Microsoft Azure (38) aims to provide an integrated development, hosting, and control cloud computing environment so that software developers can easily create, host, manage, and scale both Web and non-web applications through Microsoft data centers. To achieve this aim, Microsoft Azure supports a comprehensive collection of proprietary development tools and protocols which consists of Live Services, Microsoft .NET Services, Microsoft SQL Services, Microsoft SharePoint Services, and Microsoft Dynamics CRM Services. Microsoft Azure also supports Web APIs such as SOAP and REST to allow software developers to interface between Microsoft or non-Microsoft tools and technologies.

4.4 Obstacles and Opportunities for Cloud Computing

In addition some opinions are presented related to cloud computing, numbering the obstacles it faces nowadays, pointing out the opportunities and visions for taking as much possible advantage of the cloud.

- **Business Continuity and Service Availability**. Cloud providers can take the advantage to profit by offering availability and reliability through selling specialized hardware and software techniques at a high price. This reliability could then be sold to users as a SLA. On the other hand, to enable high availability, a good solution can be using the services of multiple cloud computing providers instead of only one, that way assuring at least one available service if other(s) went down.

- **Data Lock-in**. Since the cloud has become very popular, users are facing with the challenge to give up of the local control of their data over the storage provided in the cloud. Therefore, there are some who claim very often that their sensitive corporate data will never be put in the cloud.

  This raises concerns regarding user privacy protection because users must outsource their data, which requires a guarantee of the availability of the data files stored on the distributed servers. Moreover a problem of extracting the data from the cloud can appear.

  Generic APIs or their standardization can solve the problem of monopoly of users' data that way permitting a SaaS developer to deploy services and data across multiple cloud computing providers and prevent failure of a single company to lock-in copies of customer data with it.

- **Data Confidentiality/Audibility**. The implications of cloud computing include not only the distant nature of service provision, but also the obscurity of the particular elements involved.

  A description of distributed computing system warns,*"Cloud computing is where you can't get your work done because because a crash of a computer you've never heard of stops you from getting any work done"* - Leslie Lamport (73).

  Nowadays there is an additional risk: a machine you've never heard of pretends to be another machine you've never heard of, and steals all your data. When your critical data and your critical service provisions live in the clouds, security becomes even more complex than it is already, especially for users.

  There must be an automated way to manage and analyze security control i.e., ensure document compliance with SLAs and regulatory requirements and report any events that violate any security policies or customer licensing agreements.

  According to Berkeley, there are no fundamental obstacles to making a cloud-computing environment as secure as the vast majority of in-house IT environments, and that many of the obstacles can be overcome immediately with well-understood technologies such as encrypted storage, Virtual Local Area Networks, and network middleboxes (e.g. firewalls, packet filters). Typical example would be encrypting the data before placing it in a Cloud and may be even more secure than unencrypted data in a local data center; this approach was successfully used by TC3, a healthcare company with access to sensitive patient records and healthcare claims, when moving their HIPAA-compliant application to AWS (6).

- **Performance Unpredictability and Appearance of Bugs**. According to author's experience in (51), multiple virtual machines (VMs) can share CPUs and main memory surprisingly well in cloud computing, but that network and disk I/O sharing is more problematic. According to them, one opportunity would be to improve architectures and operating systems to efficiently virtualize interrupts and I/O channels.

  Although there are many tasks with parallelism that can benefit from elastic computing, scheduling of virtual machines can appear as another unpredictability obstacle for some classes of batch processing programs, specifically for high-performance computing.

  Related to appearance of bugs in the system, it is very common that not always they can be reproduced in smaller configurations, so the debugging must occur at scale in the production data centers.

- **Over- and Under-provisioning**. Risks of over-provisioning (underutilization) and under-provisioning (saturation) are what currently concern the cloud providers. While the cost of over-provisioning is easily measured, the cost of under-provisioning is more difficult to measure yet potentially equally serious: not only do rejected users generate zero revenue, they may never come back. (51)

- **Scalable Storage and Quick Scaling**. The appeal of cloud computing is its short-time usage, implying scaling down as well as up when demand drops, no upfront cost, and infinite capacity on demand. As a still open issue appears the opportunity to create a storage system that would not only meet existing programmer expectations in regard to durability, high availability, and the ability to manage and query data, but combine them with the cloud advantages of scaling arbitrarily up and down on demand.

  Google AppEngine automatically scales in response to load increases and decreases, and users are charged by the cycles used. AWS charges by the hour for the number of instances you occupy, even if your machine is idle.

  Another reason for scaling is to conserve resources as well as money. Since an idle computer uses about two-thirds of the power of a busy computer, careful use of resources could reduce the impact of data centers on the environment, which is currently receiving a great deal of negative attention. A fast and easy-to-use snapshot/restart tool might further encourage conservation of computing resources.

- **Software Licensing**. An obvious example for high cost of software license is SAP announcing that it would increase its annual maintenance fee to at least 22% of the purchase price of the software, which is close to Oracle's pricing.

  As a solution for this would be either for open source to remain popular or simply for commercial software companies to change their licensing structure to better fit cloud computing. For example, Microsoft and Amazon now offer *pay-as-you-go software licensing* for Windows Server and Windows SQL Server on EC2.

- **Interoperability Amongst Clouds**. E. Michael Maximilien in (73) argues that interoperation amongst cloud infrastructure and cloud application platforms is the true way to help enterprises unlock the full potential of cloud computing. This is analogous to how various network providers came together under a common set of protocols and abstraction layers to result in the Internet. Otherwise, we risk creating multiple islands of cloud solutions that will be detrimental to all involved (73).

- **Cloud as Model for Various Mash-ups**. Dave Thomas (73) sees a huge potential in the cloud computing to provide enterprises with a flexible model composed of internal and external services which combine diverse data sources and deliver them as enterprise mash-ups. While many focus on the economies of scale of platform as a service the author foresee major opportunity in innovative programming models which empower business application development and delivery, leverage the cloud and avoid wrapping simple cloud services in complex object frameworks.

  As he states further on we are only beginning to understand how to deal with massive volumes of data in a world where programmers will be doing more searching and navigation than computation. Many software professionals, claims Tomas, have only a superficial knowledge of SQL and little experience with complex functional or deductive queries.

  "We need to make these high barrier language concepts accessible via end user tools. Security and privacy demand new solutions to provide businesses and consumers confidence that their data is secure and private. Trusted, robust environments require significant improvements in both software and hardware platforms" (73).

4.5 Cloud Computing Security Overview

Since the cloud has become very popular, users are facing with the challenge to give up of the local control of their data over the storage provided in the cloud. There are some who claim very often that their sensitive corporate data will never be put in the cloud.

In cloud computing, a data centre holds information that end-users would more traditionally have stored on their computers. This raises concerns regarding user privacy protection because users must outsource their data, which requires a guarantee of the availability of the data files stored on the distributed servers. Moreover, current cloud offerings are essentially public (rather than private) networks, so the system is exposed to more attacks.

It is very important to recognize effectively corruption of a data and its modification by unauthorized entities as a result of server being compromised. Furthermore detecting the source server that contains the error is very important after the inconsistencies are localized. Data redundancy can be employed with technique of erasure-correcting code to further tolerate faults or server crash as user' data grows in size and importance. In some cases, the user may need to perform block level operations on his data.

Cloud services should preserve data integrity and user privacy. At the same time, they should enhance interoperability across multiple cloud service providers. In this context, it must be investigated further data-protection mechanisms to secure data privacy, resource security, and content copyrights.

4.5.1 Security Related Opinions

Large companies and cloud providers have already published their white papers and technical documents expressing their attitudes for cloud computing as well as suggesting various ideas of how to face an upcoming security related threats.

According to Berkeley, there are no fundamental obstacles to making a cloud-computing environment as secure as the vast majority of in-house IT environments, and that many of the obstacles can be overcomed immediately with well-understood technologies such as encrypted storage, Virtual Local Area Networks, and network middleboxes (e.g. firewalls, packet filters). Typical example would be encrypting the data before placing it in a Cloud and may be even more secure than unencrypted data in a local data center; this approach was successfully used by TC3, a healthcare company with access to sensitive patient records and healthcare claims, when moving their HIPAA-compliant application to AWS (6).

Another approach to protect in the cloud could be adding the audibility as an additional layer beyond the reach of the virtualized guest OS (or virtualized application environment), providing facilities arguably more secure than those built into the applications themselves and centralizing the software responsibilities related to confidentiality and audibility into a single logical layer. Such a new feature reinforces the cloud computing perspective of changing our focus from specific hardware to the virtualized capabilities being provided.

Despite the cloud's huge potential in reduced costs and improved productivity and overwhelming enthusiasm from customers, security experts repeatedly warn that security problems could inhibit wide adoption of the cloud model. For example, John Chambers, CEO of Cisco, stated in his keynote speech at the 2009 RSA conference, "cloud computing is a security nightmare, and it can not be handled in traditional ways" (138).

Another view describes the cloud computing as a platform for Malice Cloud providers that implicitly have access to growing amounts of data and to processes that work on data of all kinds (54). The article makes very good security observation from the aspect of the cloud providers, the users, the crafty users - "frenemies" in the cloud who extract new information by-products of retail cloud innovations.

4.5.2 Laws and Legal Rights

A big concern in cloud computing is that many nations have laws requiring SaaS providers to keep customer data and copyrighted material within national boundaries. Similarly, some businesses may not like the ability of a country to get access to their data via the court system; for example, a European customer might be concerned about using SaaS in the United States given the USA PATRIOT Act. (1).

The cloud gives the SaaS providers and SaaS users greater freedom to place their storage. For example, Amazon provides S3 services located physically in the United States and in Europe, allowing providers to keep data in whichever they choose. With AWS regions, a simple configuration change avoids the need to find and negotiate with a hosting provider overseas.

The Google App Engine's terms of service require the user to "agree that Google has no responsibility or liability for the deletion or failure to store any Content and other communications maintained or transmitted through use of the Service" (19). Amazon Web Services' terms of service make clear that the company has "no liability to you for any unauthorized access or use, corruption, deletion, destruction or loss of any of Your Content or Applications" (7). This is a real example that reflects the still not well determined cloud security agreements and policies.

4.5.3 Security Threats

The could affect the privacy and security of users' interactions. Security threats might happen in resource provisioning and during distributed application execution. There is also possibility for new threats to emerge. For instance, hackers can use the virtualized infrastructure as a launch pad for new attacks.

Security threats faced by cloud data storage can come from two different sources. On the one hand, a cloud service provider can be self-interested, untrustworthy and possibly malicious. Not only does it desire to move data that has not been or is rarely accessed to a lower tier of storage than agreed for monetary reasons, but it may also attempt to hide a data loss incident due to management errors, Byzantine failures and so on. On the other hand, there may also exist an economically- motivated adversary, who has the capability to compromise a number of cloud data storage servers in different time intervals and subsequently is able to modify or delete users' data while remaining undetected by cloud service providers for a certain period (132).

According to (54), the clouds are good front for hackers and that if users choose providers according to the service, hackers can then do the same. Also justified support of malicious networks by the ISPs, enables the same possibility for the cloud providers. They say that even the law itself might be a kind of enabler. Therefore, if copyright holders can invoke the law to scan a cloud for content that violates their rights, other kinds of scanning might occur by third parties as well. "Clouds can be terrific havens. The intercloud - cloud of clouds - promises to recapitulate the experience of the Internet as the network of networks. Still, global cloud systems will tempt global criminal enterprise."

Figure 4.3 depicts security of cloud computing as the biggest challenge of the cloud on-demand model.

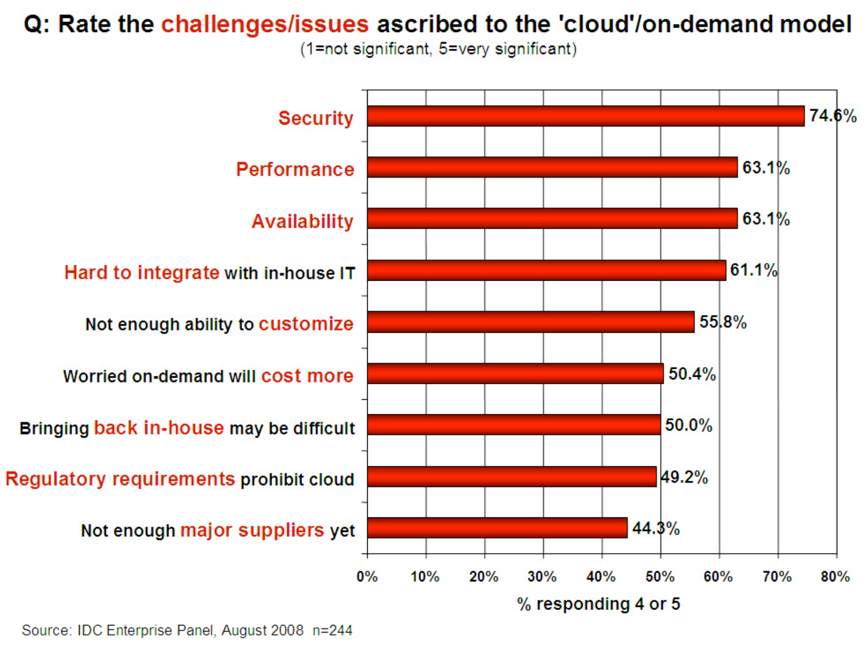


Figure 4.3: On-demand Cloud Model Challenges

4.6 Security Issues and Types of Risks in Cloud Computing

An important point to keep in mind when moving to the cloud is that the cloud does not introduce any new security threats or issues. To put security in perspective, cloud computing as a whole can be considered the ideal use case to highlight the need for a consistent, transparent, standards-based security framework regardless of cloud deployment model. As companies move or build solutions in the cloud, having this consistent security model is vital to simplify development and to avoid vendor lock-in and preserve their IT investments.

The most significant difference when considering security from a cloud perspective is the enterprise's loss of control, as opposed to any particular technical challenge. With an in-house application, controlling access to sensitive data and applications is crucial. With a cloud-based application, access control is just as important, but the infrastructure, platform and application of security is under the direct control of the cloud provider.

4.6.1 Security Issues

There exist several security topics for cloud computing. They are all well synthesized and described in (76).

- **Regulations**. Regulations are not technical issues, but they must be addressed. Laws and regulations will determine security requirements that take priority over functional requirements.
- **Security Controls**. Although a given consumer might need all of these security controls, consumers should be wary of any cloud provider that makes security-related claims and reassurances without an infrastructure capable of delivering all of them.
- **Security Federation Patterns**. To implement these security controls, several federation patterns are needed. Cloud providers should deliver these patterns through existing security standards.

4.6.1.1 Regulations

Beyond all of the technical issues in using cloud computing is the harsh reality of regulations. For a variety of reasons, governments around the world are concerned about the use of cloud computing. Many countries have strict privacy laws that prohibit certain data from being stored on a physical machine located outside that country. There are often stiff penalties for organizations (and, in some cases, their executives) that violate those laws. Any organization storing sensitive data in the cloud must be able to prove that their cloud provider never stores that data on a physical server outside a particular geographic area.

In addition to government agencies, many trade and industry groups create regulations as well. While those regulations might not be required by law, they represent best practices. Similar concerns apply to applications running in the cloud. If a virtual machine is running in the cloud, can an application running on that VM access sensitive data? This is a grey area that many countries have not addressed yet, although new laws and regulations will be created on an ongoing basis. Following these laws and regulations will take precedence over all other requirements. A new law might require an organization to spend their resources changing an applications' infrastructure instead of adding features to it. These changes should be managed and alerted for new laws and regulations as they emerge.

4.6.1.2 Security Controls

A number of controls are necessary to adequately secure any system. The following subsection describes those security controls as they have been given in (76)

- **Asset Management**. It must be possible to manage all of the hardware, network and software assets (physical or virtual) that make up the cloud infrastructure. This includes being able to account for any physical- or network-based access of an asset for audit and compliance purposes.
- **Cryptography: Key and Certificate Management**. Any secure system needs an infrastructure for employing and managing cryptographic keys and certificates. This includes employing standards-based cryptographic functions and services to support information security at rest and in motion.
- **Data / Storage Security**. It must be possible to store data in an encrypted format. In addition, some consumers will need their data to be stored separately from other consumers' data.
- **Endpoint Security**. Consumers must be able to secure the endpoints to their cloud resources. This includes the ability to restrict endpoints by network protocol and device type.
- **Event Auditing and Reporting**. Consumers must be able to access data about events that happen in the cloud, especially system failures and security breaches. Access to events includes the ability to learn about past events and reporting of new events as they occur. Cloud providers cause significant damage to their reputations when they fail to report events in a timely manner.
- **Identity, Roles, Access Control and Attributes**. It must be possible to define the identity, roles, entitlements and any other attributes of individuals and services in a consistent, machine-readable way in order to effectively implement access control and enforce security policy against cloud-based resources.
- **Network Security**. It must be possible to secure network traffic at the switch, router and packet level. The IP stack itself should be secure as well.
- **Security Policies**. It must be possible to define policies, resolve, and enforce security policies in support of access control, resource allocation and any other decisions in a consistent, machine- readable way.

The method for defining policies should be robust enough that SLAs and licenses can be enforced automatically.

- **Service Automation**. There must be an automated way to manage and analyze security control flows and processes in support of security compliance audits. This also includes reporting any events that violate any security policies or customer licensing agreements.
- **Workload and Service Management**. It must be possible to configure, deploy and monitor services in accordance with defined security policies and customer licensing agreements.

Bellow are some of standards that can be applied to these controls. For Cryptography: Key and Certificate - *KMIP, the OASIS Key Management Interoperability Protocol* (25); Data / Storage Security - *IEEE P1619, developed by the IEEE Security in Storage Working Group* (84); Identity, Roles, Access Control and Attributes - *SAML, the OASIS Security Assertion Markup Language* (33), X.509 Certificates, part of the ITU Public Key and Attribute Certificate Frameworks Recommendations (41); Security Policies - *XACML, the OASIS eXtensible Access Control Markup Language* (42); Workload and Service Management - *SPML, the OASIS Service Provisioning Markup Language* (35).

4.6.1.3 Security Federation Patterns

Federation is the ability of multiple independent resources to act like a single resource. Cloud computing itself is a federation of resources, so the many assets, identities, configurations and other details of a cloud computing solution must be federated to make cloud computing practical. The requirements from the previous section are implemented via the following federation patterns:

- **Trust**. The ability for two parties to define a trust relationship with an authentication authority. That authentication authority is capable of exchanging credentials (typically X.509 certificates), and then using those credentials to secure messages and create signed security tokens (typically SAML). Federated trust is the foundation upon which all the other secure federation patterns are based.
- **Identity Management**. The ability to define an identity provider that accepts a users credentials (a user ID and password, a certificate, etc.) and returns a signed security token that identifies that user. Service providers that trust the identity provider can use that token to

grant appropriate access to the user, even though the service provider has no knowledge of the user.

– **Access Management**. The ability to write policies (typically in XACML) that examine security tokens to manage access to cloud resources. Access to resources can be controlled by more than one factor. For example, access to a resource could be limited to users in a particular role, but only across certain protocols and only at certain times of the day.

– **Single Sign-On / Sign-Off**. The ability to federate logins based on credentials from a trusted authority. Given an authenticated user with a particular role, federated single sign-on allows a user to login to one application and access other applications that trust the same authority. Federated single sign-off is part of this pattern as well; in many situations it will be vital that a user logging out of one application is logged out of all the others. The Single Sign-On pattern is enabled by the Identity Management pattern.

– **Audit and Compliance**. The ability to collect audit and compliance data spread across multiple domains, including hybrid clouds. Federated audits are necessary to ensure and document compliance with SLAs and regulatory requirements.

– **Configuration Management**. The ability to federate configuration data for services, applications and virtual machines. This data can include access policies and licensing information across multiple domains. Because existing best practices for security apply to the cloud, providers should use existing standards to deliver these federation patterns.

4.6.2 Risks

According to the authors of the (139) the major threats against the safety of document service and the privacy of user documents focus on two concepts:

1) documents would be intercepted and captured during transferring from client-end to the cloud and 2) access control for documents stored in the cloud.

Thus, cloud environment needs new models to handle potential security problems. This new model should allow an information owner to protect their data while not interfering with the privacy of other information owners within the cloud. From the view of client, the remote service deployed in cloud is hardly to be regarded as trustworthy in default situation.

Jinpeng Wei et al. in the paper (138) explain the new risks that face administrators and users (both image publishers and image retrievers) of a cloud's image repository.

4.6.2.1 Publisher's Risk

By publishing an image, the publisher risks releasing sensitive information inadvertently. Although traditional software publishers run similar risks, the problem is larger for image publishers because images contain installed and fully configured applications: the configuration might require dangerous operations like creating password-protected user accounts and, if the publisher sets up the application by running an instance of the image, it may unwittingly create files that should not be made public.

4.6.2.2 Retriever's Risk

The retriever risks running vulnerable or malicious images introduced into the repository by a publisher. While running a vulnerable virtual machine lowers the overall security level of a virtual network of machines in the cloud, running a malicious virtual machine is similar to moving the attacker's machine directly into the network, bypassing any firewall or intrusion detection system around the network.

4.6.2.3 Repository administrator's risk

The repository administrator risks hosting and distributing images that contain malicious or illegal content. It is as much in his interest as in the consumer's to keep the images free of such content. To address those risks, they propose an image management system that controls access to images, tracks the provenance of images, and provides users and administrators with efficient image filters and scanners that detect and repair security violations. Filters and scanners achieve efficiency by exploiting redundancy among images.

4.7 Related Work and Ideas for Secure Clouds

Although still novel, the cloud paradigm has provoked number of research works. From investigating its architecture and the way it could interact with other platforms, cloud computing security has been identified by many research works as a very crucial and still not well explored area.

Results shown in (143) bridge security-related gaps between industrial views of clouds and that from theoretical researchers. To demonstrate interesting applications of the result, parallel computing of MapReduce over Clouds is considered. They give formalization of cloud computing claiming that "Secure Cloud computing can be viewed as an extension of classic secure computations of any function in the computational setting".

The mechanism of security document service for cloud computing environment has been proposed and an archetype has been given in (139). In this mechanism, content and format are separated from document to keep their privacy. Also, an optimized authorization method has been proposed for assigning access right of document to authorized users.

To ensure cloud data storage security, (135) claim that it is critical to enable a third party auditor (TPA) to evaluate the service quality from an objective and independent perspective. Public verifiability also allows clients to delegate the integrity verification tasks to TPA while they themselves can be unreliable or not be able to commit necessary computation resources performing continuous verifications. Another major concern is how to construct verification protocols that can accommodate dynamic data files. In the paper, a problem is explored of providing simultaneous public verifiability and data dynamics for remote data integrity check in cloud computing. It is designed to meet these two important goals while efficiency being kept closely in mind which offers a verified fully dynamic data operation.

Related to ensuring correctness of users' data in cloud data storage, Cong Wang et al. in (132) proposed an effective and flexible distributed scheme with explicit dynamic data support, including block update, delete, and append. They rely on erasure-correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability.

By utilizing the homomorphic token with distributed verification of erasure-coded data, the scheme achieves the integration of storage correctness insurance and data error localization, i.e., whenever data corruption has been detected during the storage correctness verification across the distributed servers, it can be almost guaranteed the simultaneous identification of the misbehaving server(s). Through detailed security and performance analysis, is shown a scheme that is highly efficient and resilient to Byzantine failure, malicious data modification attack, and even server colluding attacks.

A paper name "Privacy as a Service" (85) presents PaaS, a set of security protocols for ensuring the privacy of customer data in cloud computing infrastructures. The security solution relies on secure cryptographic coprocessors for providing a trusted and isolated execution environment in the computing cloud. Moreover, PaaS provides a privacy feedback process which informs users of the different privacy operations applied on their data and makes them aware of any potential risks that may jeopardize the confidentiality of their sensitive information.

Three privacy models are distinguished: compliance-based trust, full trust and no trust. The idea is that before uploading the data to be stored and processed in the cloud, the customer classifies the data based on significance and sensitivity into 3 privacy categories: No Privacy, Privacy with Trusted Provider and Privacy with Non-Trusted Provider. The paper also discussed the PasS protocols and describes the privacy enforcement mechanisms supported by them.

In the journal "Can a Trusted Environment Provide Security?" (93), is described a new idea for *security as a service*, from an aspect of the users of the cloud, claiming that in order for security to become a cloud offer, customers have to establish required security policies and risk framework, i.e. the cloud users are the one to identify, assess, measure and prioritize their system risks.

In the cloud world, the customers are the only responsible for their SaaS running in the cloud platform, i.e. they have to purchase, install and configure the antiviral, antispyware, antimalware services in their host environment. They claim however that if the cloud providers offered security as a service in their virtualized environments, customers could integrate these security measurements into their risk profile to ensure that the cloud risk posture is acceptable.

There exist several other researches, technical reports and white papers in relation with cloud computing security. They threat the security form different perspectives purposing various interesting solutions of how to cope with security threats, some of those are (59; 65; 68; 83; 90; 91; 92; 98; 109; 134).

Chapter 5

Hybrid P2P and Cloud Computing Model

5.1 Introduction

Soon after P2P file sharing market reached the fame, a new way of distributing the content was introduced - streaming (realtime and on demand) on the top of the P2P overlay that attracted many users to switch to a new way of watching the media content with no need to download. Today, Cloud Computing (CC) has placed a serious strain on the business market. In the same time, it inspired researches to think of integrating an already existing technologies with cloud computing, taking the advantages CC offers.

M. Fouquet et al.(72), present a good overview of the utilities CC offers. They describe a distributed application-layer multicast for video streaming engined by relay servers, discussing its possible integration with the cloud. They present in details the distribution trees topology, numbering various use cases of how to benefit from the cloud. Latency and bandwidth issue in video streaming is mentioned without further details.

Therefore as principal objective in our work, we consider building automatic API functions based on Quality of Service (QoS) parameters to leverage a novel hybrid cloud and P2P architecture for multimedia streaming. Cloud paradigm is favored to be future key for bringing back the Client-Server (CS) topology in multimedia communication, and by expanding it for P2P streaming support it is believed it could bring double benefit to both the cloud service providers, and the end users.

The architecture presented in this chapter[1] offers a transparent approach towards QoS guided business model, relaying on cloud scalability and robustness. Organized as a multimedia streaming distributed overlay, it presents an idea for application interface for providers of multimedia streaming content. The Isabel (24) platform for videoconferencing service includes a feature in its infrastructure, that registers and stores streaming packet time stamps (incoming and outgoing) of the participants in a video conference.

By extending this feature it is possible to implement automatic functions for calculation of QoS parameters-such as bandwidth, jitter and latency. The functions would be part of a web service and will be represented to the cloud providers' clients through a user friendly interface. This would enable them by connecting to this provider's page, to consult the current status of a streaming content, the ID of the connected clients using the service and their QoS status. The API would furthermore offer a price model, that permits straightforward decision on whether to watch streaming in a CS or P2P manner. Unlike the models for ISP guided choice of peer selection, we suggest a new and completely independent model for selection guided by clients' preferences for QoS parameters and price packets. This model satisfies the clients as they could take the maximum advantage of this model guaranteed with QoS parameters, while the provider has complete monitoring of the dynamic QoS changes, therefore could react on time to reinforce its resources for better scalability.

5.2 Related Work

No doubt exists of the achievement P2P technology has brought, as many applications, topologies and protocols have marked its maturity over the years. Implemented as an organized overlay in order to overcome the CS based limitations for expensive bandwidth provision cost, P2P attracted various open source and commercial designs for content sharing, Video on Demand (VoD) and live streaming. These have proven P2P's model increased contribution for scalability and robustness. As pointed out in Section 2.1, BitTorrent (62) was the pioneer in this area and is still number one among the P2P file-sharing applications.

Cloud computing brings back in game the virtual centralized model. The scalability and robustness of the cloud infrastructure offered by leading cloud providers as Amazon, Google, IBM, Oracle, leverage various commercial applications to move in the cloud. Multimedia live streaming and VoD Software as

[1] *The presented architecture is part of a paper called "A Novel P2P and Cloud Computing Hybrid Architecture For Multimedia Streaming With QoS Cost Functions" submitted to ACM Multimedia 2010 Conference*

a Service, (SaaS) (28; 40) have already started to give their bet over the cloud infrastructure, in this case Amazon EC2 (3) and Cloud Front (2).

Cloud computing profit oriented nature could render more difficult the establishment of a mutual negotiation policy when combined with other non-profitable platforms. Such an example for dealing with the long lasting battle between the P2P overlay networks and the ISPs have been described in (45; 125; 133).

Based on locality-awareness method, overlay topology construction approach or on both, in order to reduce the cross-ISP traffic they suggest various methods such as oracle web service offered by the ISP, adaptive peer selection algorithm, algorithm based on a gossip protocol or hybrid approach adapting between biased and random peer selection. These techniques aim to facilitate a P2P node to choose neighbors independently based on a certain criteria. Last example presents an algorithm for peer-assisted VoD in which peers bias transmission rates to their neighbors, and dynamically adjust these rates.

The research work mentioned so far treats the trade off between the ISP and the P2P overlays build on the top of their network. However comparing among commercial cloud applications, no ISP's uniform payment policy for multimedia streaming and VoD service exist. Offers vary depending on the required cloud disc space or CPU for PaaS and IaaS solutions, while SaaS providers define their own cost models depending on the services they offer.

5.3 Architecture Overview

Cloud commercial multimedia streaming applications are expected to meet certain QoS parameters. Yet no concrete pricing model based on automatic QoS parameters exists for commercial use. We wanted to explore this possibility and define price packets for customers of multimedia streaming service by considering three type of QoS parameters such as jitter, latency and bandwidth.

A hybrid architecture is proposed for multimedia streaming that offers API functions based on the above mentioned parameters. Furthermore, the price politics for the commercial CS part of the architecture are put on scene with an alternative way of the non profitable P2P part.

The architecture takes advantage of the CC scalable infrastructure that facilitates deployment of stable and robust services. An idea for a SaaS is offered to the users of the cloud infrastructure such as Autonomous Servers (AS) or ISPs, that could be deployed as a web service application on the cloud.

Figure 5.1 represents a general view of the proposed architecture. It depicts the cloud that contains multimedia streaming servers, (their number depends on the provider of the streaming service with possibility to scale as the number of customers or petitions rise). The service has *first level* or directly connected clients (Client C1, C2 , C3,..C6) and *higher level clients* (Client C4_1, C4_2, C5_1 and C5_2). The idea behind this is that first level clients after login, consult and choose one among the three types of price packets. Afterwards, they decide to make contract directly with the provider of the service enjoying higher streaming quality.

Similarly, higher level are clients that by consulting API functions obtain an information list with QoS status for all connected clients and have decided to which client to connect. By doing this they agree on lower streaming quality (determined in the price model for higher level customers) and contract the first (or higher level) customers instead of the provider. There exists possibility for peers to connect to higher level clients who want to offer their service for free (Peer P1, P2, P2_1, P3).

This way the streaming topology is organized in a tree-based manner but it does not exclude the possibility to adapt it for a mesh-based overlay. However, discussing types of overlay topologies are out of the scope in this paper.

The service provider has direct centralized management of the contract politics among all type of customers. It also takes control over the streaming and stored content on the server with possibility to contract external streaming server for single or short term streaming. This may appear as a result of increased petitions for live or on demand streaming content.

Such a cooperation with the service provider acting as a mediator among third party servers and own clients, contributes the business as more streaming contents become available. The owner of an external streaming server profits from the cooperation with the cloud service provider and indirectly disposes with the same user friendly interface which attracts a number of new clients. Also it uses a stable service that relies on a scalable cloud computing platform disposing with higher bandwidth, lower latency, better load balancing, scalability and robustness. Aiming to figure out possible cost models between various external servers and SaaS providers on the cloud could be possible research work that could emerge from this idea.

Following the idea of Amazon Reserved Instances (4) and payment methods some telecommunication companies offer for various services, a cost model is introduced, offered by the cloud provider of the streaming service to the end users. It is foreseen that it could possibly serve as a base to build uniform model for coordination among various service providers of multimedia streaming, on the one hand and cloud provider companies on the other hand. Such a user friendly price model will enable deployment of various SaaS in the cloud that can profit out of the dynamic QoS parameters displayed for all subscribed clients, and would permit to negotiate price among different clients.

In continuation the APIs functions will be described following the event flow scenario.

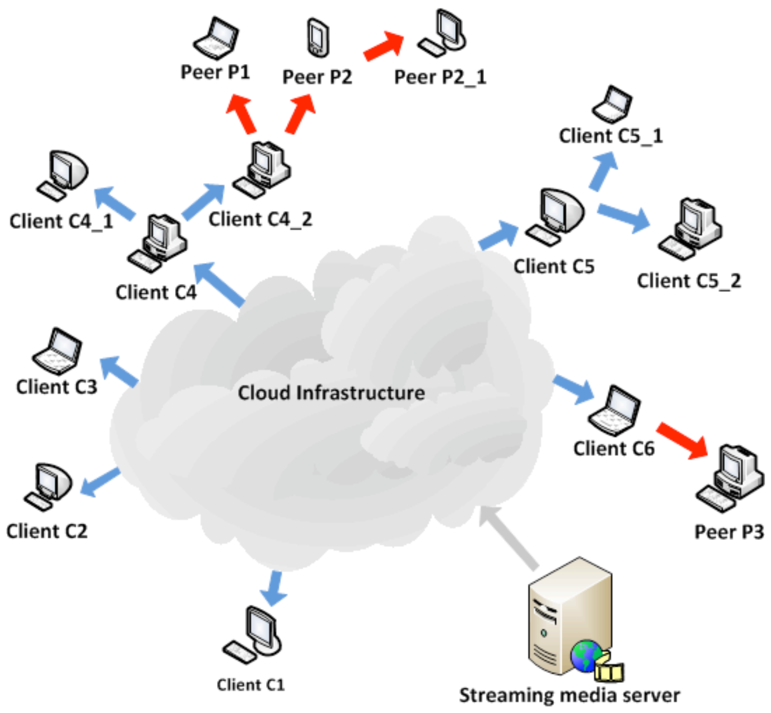


Figure 5.1: CS-P2P Streaming Cloud Architecture

Users login to the service provider page (web service), willing to watch some streaming program or content on demand.

```
(1) get_user_table(streaming content)
(2) get_price_packets(streaming content)
```

After choosing a content, the user invokes web service API function (1) that returns a table with all currently subscribed clients watching the `streaming content` specified in the function. Call of function (2) would return a shorten example table as shown in continuation, with type of price packets and users of each packet for the specified `streaming content`. By the ID of each user are shown the three values of the QoS parameters for video streaming packets.

QoS	**UserID: BV(Mbps), LV(ms), JV**		
Gold	C1: 25, 20, 0.0435	C8: 19, 10, 0.5366	C4:..
Silver	C5: 13, 40, 1.4785	C9: 11, 60, 1.3299	C6..
Bronze	C2: 4, 80, 2.3401	C5: 5, 90, 2.7832	C7:..

Price packets are defined regarding the QoS bounds guaranteed to the user of the streaming service. We suggest a generic model containing three types of price packets, Gold, Silver and Bronze.

Gold packet is most expensive offering best quality streaming with the highest bandwidth and least possible jitter and latency. **Silver** and **Bronze** offer lower quality specifically adapted for users who do not insist on the highest quality, but could rather tolerate lower QoS at a lower cost. The service could be set up on demand in minutes, hours, per streaming content etc., which together with the price of the packets is determined by the providers.

Besides direct CS connection, clients are allowed to connect to other clients and extend the streaming tree. New users can obtain information for connected clients invoking the associated API functions. Calling (3), would return all the first level clients of the streaming content and their current QoS values, while calling (4) will show second, third etc., level clients. One can require a view of all peers registered in the session by calling (5).

```
(3) get_direct_clients(streaming content)
(4) get_other_clients(streaming content)
(5) get_peers(streaming content)
(6) get_bandwidth(streaming content)
(7) get_latency(streaming content)
(8) get_jitter(streaming content)
```

APIs (6, 7, 8) give a list of all clients and their current values for bandwidth, latency and jitter respectively. It could be also possible to combine API functions or specify desired values for some parameters.

```
(9) get_bandwidth_and_latency(streaming content)
(10) get_peers(bandwidth, jitter, latency)
(11) get_direct_clients_of_packet(bandwidth, jitter,
     latency, packet)
```

Function (9) returns a list of clients together with their bandwidth and latency values, while (10) lists all peers who share the specified values for `bandwidth` and `latency` (some of those could be left empty if not required). (11) is complex function returning list of direct clients with specified QoS who have contracted price packet `packet`, where packet value can be gold, silver or bronze.

Having this information a new user can choose, either to connect to the provider and become its first level client by choosing which type of price packet to subscribe, or connect to other client in the streaming tree depending on their

QoS and price preferences. If none of those it could simply connect as peer and start streaming for free. Once the choice is made it could notify the provider, who takes care of the payment method.

With this, potential clients and peers have transparent view of the status of some streaming content with respect to the QoS type, their value intervals, the prices assigned to each interval including the clients and their status. This allows them, to choose depending on the QoS values, the desired type of contract or connection.

```
(12) get_clients_status(streaming content)
```

Another API function (12) would return a table with the status of clients sorted according to the service they offer. Results of this call for only one user are shown in the following table:

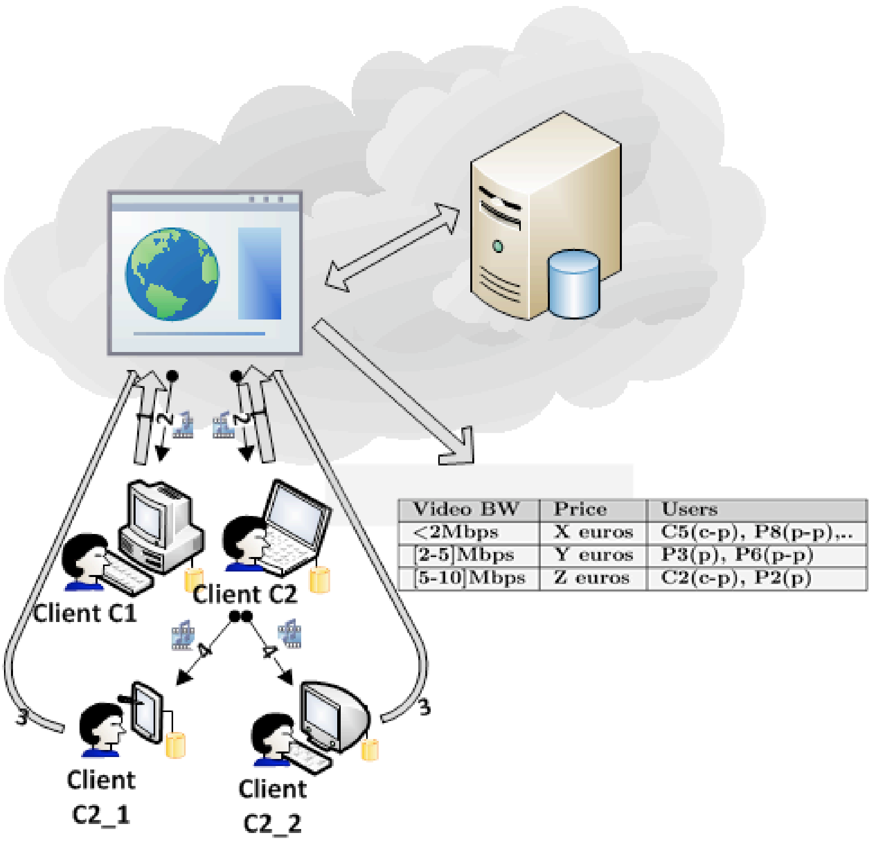
QoS	**C**	**C-C**	**C-P**	**C-C,P**	**P**	**P-P**
User ID	C1	C9	C5	C2	C10	C12
B(Mbps)	25	11	5	4	2	1
L(ms)	20	60	90	80	90	110
J(ms)	0.04	1.32	2.78	2.34	3.03	3.44

The first row contains letters to denote possible status combinations. The first letter in a tuple denotes if user identified with its ID number is client or peer, and the second one denotes whether they permit their service to other clients, peers or both. In a case they do not offer further streaming, the second letter is dropped. For example $C8(c - c)$ denotes user $C8$ is client and permits only clients connected, or $C3(c - c, p)$ - user $C3$ is client and permits both clients and peers connected. Note that there is no category $ID(p - c)$ for obvious reasons that once an user becomes a peer, it is not permitted neither logical to offer its service to clients, due to their low QoS values. Anyway this is not allowed since no price is established for very low QoS and hence, a client or peer with such values for QoS parameters could offer service only to peers further in the P2P tree.

On the other side it could be feasible by the third, fourth etc., level clients, i.e., sub-users of silver or bronze packets, to offer their service for further P2P redistribution of the streaming content due to low QoS values, that are not enough to offer streaming services to other clients. For example, lets say that the client C of bronze packet has the QoS values for video (B: 2Mbps, L: 90ms, J: 2.55);

this user has no befit to charge another client(s) for its service therefore sets up his status to $C(c-p)$ extending the tree with P2P streaming only.

Each client/peer decides their own status. As noted earlier the goal is to union CS and P2P streaming under the same model, so that everybody can benefit from the architecture. Once a choice is made the provider assigns the client an ID number, they pay for the service, set up the status and start streaming.



Video BW	Price	Users
<2Mbps	X euros	C5(c-p), P8(p-p),..
[2-5]Mbps	Y euros	P3(p), P6(p-p)
[5-10]Mbps	Z euros	C2(c-p), P2(p)

Figure 5.2: Example Topology with QoS Table

Figure 5.2 depicts reduced streaming scenario including a table as a part of the interface that contains information classified according to the bandwidth intervals. Each column contains QoS value interval as upper and lower bounds of the bandwidth and the price assigned to that interval. Below each price is a view of all currently connected clients under that category together with their status. If new user finds an interval and its assigned price convenient, it chooses to which client or peer of that category to sent petition for connection, depending on their current QoS values.

The cloud architecture as described enables better centralized control of the involved clients and peers. For example, in order to avoid abuse of the service and prevent various clients to offer the service to other clients setting personal random prices, all contracts are established through the cloud control system. Payment between new and a current client will be purchased through a web service payment method after login. At the same time the QoS interface reflects current state of the resources which permits the service provider online control and monitoring with possibility if scalability requirement appears, to turn on or contract new cloud servers. This would reduce latency and contribute higher robustness.

As for the churn issue, once a client pays the service it is in their best interest to stay connected until the end of the streaming due to the payment acquisition. However, legal commitments should be specified in the SLA for higher level clients who want temporally or permanently to leave the session. The service should then redirect the affected clients to other clients with status type $ID(c-c)$ from the same price rang. Churn among peers does not directly affect the global integrity of the architecture.

Currently it is very difficult to combine client-server and peer-to-peer streaming within the same service, mostly because of cost policies that require accurately established model. It is believed that this is the first attempt to combine these two technologies under the umbrella of the cloud because we recognize high potential in the cloud infrastructure, especially regarding precise cost models that would facilitate future reinforced interaction of end users and cloud service providers.

5.4 Calculus of QoS Parameters

The QoS values mentioned in the previous section, are based upon an automatic calculation algorithm to be deployed. We base the packet time stamps acquisition on a feature in the Isabel platform for videoconferencing service and extend it to deploy the described APIs. The idea is described on Figure 5.2. As a streaming session initiates, time stamps of outgoing and incoming audio and video packets are registered. The received data is stored on the provider database server in the cloud and also on the clients' machines. Only time stamps of outgoing packets will be considered assuming the processing packet time very small, thus irrelevant for the calculation. Following are the well known formulas for QoS calculation as they would be used in our scenario, provided we have included the time stamp feature in the web service logic:

We observe Client C2_2 on Figure 5.2. If t_{C22} is packet time stamp at Client $C2\_2$, then its latency will be calculated by summing all the previous latencies from the server up to client $C2$ in the tree, adding the new time stamp for Client

$C2\_2$, or $latency(C2\_2) = t_{C22} + latency(C2)$, where $latency(C2)$ is the latency of client $C2$ for the same packet. Latency for both audio and video packets is calculated according to the same formula and so for every new user in the streaming tree.

The jitter represents the statistical variance of RTP data packet inter-arrival time. Therefore if Si is time stamp for packet i, and Ri is the time of arrival in RTP time stamp units for packet i, then for two packets i and j we have the difference $D(i, j) = (Rj - Ri) - (Sj - Si) = (Rj - Sj) - (Ri - Si)$. The inter-arrival jitter of any client is calculated instantaneously as the data packets are received by the client using the difference D for that packet i and the previous packet $i - 1$ according to the formula $J(i) = J(i-1) + \frac{(|D(i-1,i)| - J(C2))}{16}$. The division by 16 is in order to reduce noise.

Finally to calculate the bandwidth we assume that the streaming packet size is constant during the streaming session. Supposing that the streaming channel is asymmetric we will consider only the upload bandwidth as of interest for new users. If the streaming bit rate br is known, depending on the audio/video codec used, the bandwidth of Client $C2$ is simply calculated as $bandwidth(C2) = n * br$, where n is the number of clients to which peer $C2$ is forwarding the streaming packets (two in this case). The bandwidth can be also calculated as $bandwidth(C2) = (ps_a * n * p_a/t_a) + (ps_v * n * p_v/t_v)$, where ps_a and ps_v are packet size of the audio and video the stream respectively (ex. bits/packet). Analogically p_a/t_a and p_v/t_v are the audio and video packet rates (ex. packet/time).

The web service registers every time stamp value and assigns it to the equations. This way, will be achieved a dynamic and more accurate QoS representation.

Chapter 6

Voluntary Computing Paradigm

6.1 Introduction to Voluntary Computing

There exist various scientific areas, primary in the field of simulation y data analysis, where the computational load of some tasks if very high, so that one computer with a standard processing power would delay with decades in executing that information.

The environment dedicated to resolve such type of tasks is called HPC (*High Performance Computing*), that normally resolves this type of problems mediate two approximations. On the other hand with an use of supercomputers, individual machines with a very high computing power, high cost and consume. On the other hand a distributed computing has appeared based on the use of clusters of machines with minor power.

The volunteer computing appeared in the middle of the nineties, inside the area of distributed computing. In this period, the personal computers started to be generalized, to increase the power and most of all, started to interconnect in a global network - the Internet. Out of here appeared the idea to use this mutual connected power. Concretely, the voluntary computing started with use of the entire unused existing processing capacity by the personal computers, which this way offer voluntarily and free of charge this exceeding capacity.

In order to be adequate to this type of processing, the executing tasks should not have a critical deadline, rather their success should depend on the capacity of the processed data. For example the first project that used this system was dedicated to search for numbers with a certain math characteristics. It was not supposed to achieve a concrete quantity of numbers in a certain time, but its success was measured depending from the quantity of numbers found and analyzed.

6.1.1 History

The first projects that are still active although with a smaller enthusiasm then when they started some years ago are **GIMPS** and **distributed.net**, lunched in the years 1996 and 1997 respectively. From one side GIMPS (17) was dedicated to search Mersenne prime numbers (26), and from other side, distributed.net (13), which initially was dedicated to break cryptographic keys mediate brute force mechanism and currently is dedicated to search mathematic series of optimal Golomb rulers (OGRs), (explained here (115)). The importance of these projects wasn't that much of their results rather their success showed that it was plausible the use of distributed computing with scientific goals.

However, in 1999 two projects appeared that received high recognition which they still have today. **University of California-Berkeley**, dedicates at analysis of information received from a radio-telescope Arecibo who searched for artificial (or "intelligent") radio signals (34; 49). **Folding@home**, deployed by the University of Stanford, dedicates to determine how the proteins are formed from the genetic sequences in the human bodies (15; 96).

There projects were the first to be recognized and followed by a more global audience, the key factor for success of distributed computing. Moreover, both developed their own middleware tools, which facilitated the distribution of the work among different users and the execution of client programs in such way that they don't interfere with the rest of the tasks of the voluntary computers. Also web sites were launched where the volunteers could register and these became an authentic social communities with extreme support of intensive use of the programs (34; 96).

Furthermore in 2002 the same group that developed SETI@home also created **BOINC**, a fundamental tool for the generalization of projects from this type, that will be explained more in details in the appropriate paragraph. To sum up, BOINC is a platform that permits any type of projects that requires the use of a distributed computing to be deployed over it.

6.1.2 Social Characteristics

The voluntary computing depends a lot on the people. By no means based on hired technical means, but on available resources of a voluntary and unpaid type, the human factor is crucial. How to motivate people to participate in projects, has become basic subject from the beginning, both within this project as in any other based on voluntary and free development (95). The main motivation that people are recognizing in these projects include:

- **Scientific Interest:** share the objective of the project is usually the main reason for which a volunteer would share its computer. Projects such as search for a cure to disease, or searching an extraterrestrial life have a strong interest for the general public (or at least for part of it).

- **Participate in a Community** has already been proven form the part of a social network, to be a great motivation for many people, in addition to the competitive factor that includes system of credits assigned to a person for the capacity assigned to a project, has increased the interest of many people.
- **Evaluation of a Machine:** many people interested in the performance of their own machine(computer), find of interest the voluntary computing as a form to make public its characteristics or as a simple method for comparison to other user's machines.
- **Economic Interest:** so far it doesn't exist a project that is commercially viable, requesting the end-user exchange in money for the processing power. The main reason for this is the lack of control of the processed data, which makes large companies or governments very reluctant to use it in commercial or critical projects.

  Therefore, in order for a project to have a success, its objectives have to be clearly explained, how the program works, the amount of progress that has been achieved and to also keep an updated information for the continuous results, both on a global and individual level. Advertising is crucial for these projects, which can be promoted a lot mediate the new means of social communication like Facebook or Twitter.

  Another important point to consider is the high possibility that a user give a false information, so the systems must have methods to evaluate and correct the received results.

6.1.3 Technical Characteristics

The volunteering computing has some characteristics that makes it very particular and provoke the existence of a series of potential problems that have to be taken into account when designing this type of projects:

The heterogeneity of computational sources, most of all talking at a level of hardware, software and availability. Every user poses a different system with a different availability, which obliges the project to be capable to adopt the workload to fit various computers. Not only this, but there exist today variety of devices with processing capacity susceptible for use (tablets, video-consoles,...) which implies to a multiple hardware and more importantly multiple operating systems. Each software requires its own version of a program, and it will be executed in a different manner depending on the hardware of the machine with adapted load. This problem can find its solution or at least bigger part of the solution in the virtual machines, like java, that would permit the functioning of a same program with any hardware support which has the virtual machine already installed.

It has already been intended to develop a tool in MIT for voluntary computing over the web with java tools, although latter it was abandoned (121). Recently it has been intended to take the advantage of the virtualization of machines, a very useful technology that permits a multi-platform for the applications of various projects, a key factor when it is not always easy or not always exist the resources to rewrite the code for different models of machines. By using virtualization, one virtual image of the entire application could be executed in the client machine. **CERNVM** is a more advanced project in this aspect (9), used for the different projects of LHC and its integration in BOINC is already very enhanced (122).

Moreover, everything possible has to be tried to ensure the tasks performed in parallel or programs scheduled at short distance are not dependent on each other, because we will never have the certainty that a particular task is to take place before a determined time, and it would be highly inefficient to have stopped a project because a machine is disconnected due to an unresolved problem with one of its parts.

There is the possibility that some user send wrong or falsified data, as was already indicated earlier. Normally this is solved on a base of redundancy (each packet of data is sent several times to analyze and only are considered as valid the results if the same answer is obtained from different sources), but this reduces greatly the system performance. Some systems have implemented more advanced measures, including a dynamic rating of the users, where duplicates are not sent to the most reliable, but rather to other less reliable. One of these algorithms is explained in (113).

In this sphere of computing it is important the wide use of simulators for causes of investigation. By simulating this type of network, the scalability becomes a very complex issue. Currently, the most complex developments are enabled over *SimGrid*, a specific tool for simulation of distributed applications in distributed environments that has lower scalability problems than many other simulators (56).

The team MESCAL (Middleware Efficiently SCALable) of LIG, the Computer Laboratory in Grenoble, involved in the design of distributed efficient architectures, has developed complex simulations (67) and analyzed real environments in order to represent sets of client machines starting form their statistical properties in a realistic form (89).

Finally and as a main reason for failure of voluntary computing as a commercial platform, is the facility which a hacker may have to **access a project data**.

It is easy to read the distributed information, but it exist also the chance to gain access to the server and be able to send harmful executables to the volunteers or even possibility to create a fraudulent project with the idea to gain access to data or to machines of volunteers. Limiting the power of the programs on the machines and the servers can be a key, but still it doesn't exist a system that is completely protected, although the use of virtual machines could improve a lot the architecture. On the other hand, a user must always be familiar with the exact material it installs and only download voluntary computational programs when it fully trust in the source of the project.

One of the major if not the largest advantages of this system, which is not taken that much into account as its processing capacity, is the cheap maintaining and automatic upgrade. Currently, the replacement of personal computers for others quite modern and powerful is very common, so often the PCs use newer technologies, far more powerful then the computers used in the universities and the laboratories. For the case of latest generation consoles, this is even more clear. For example right now they are the first to commercialize the most powerful GPUs units, with a high potential in this field. Moreover the maintenance and the expense of the park are brought by volunteers, which leads to even greater savings.

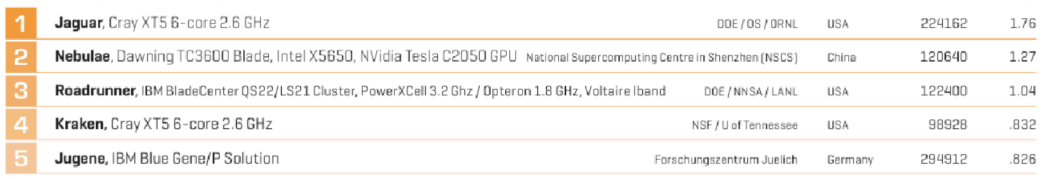
6.1.4 Present Situation

It is now believed that there are almost a million machines voluntarily participating in projects of this nature (although it is a difficult quantity to measure) that together provide about 10 PetaFLOPS (FLOPS, floating point operations per second) (48). The largest of these communities is the Folding@home, which currently holds a processing capacity of more then 6 PFLOPS alone and was the first project in history to cross the barriers of 1, 2 , 3, 4 and 5 PFLOPS (48).

Part of its success lies in the use of the processing power of the current GPUs, especially after signing a deal with Sony, who developed a specific software for their consoles PlayStation 3 to use that way the power of its processors *Cell* and GPU units in this project (16). The high processing power of GPUs is clear when it is estimated that now they account for around 70 % of the processing capacity in these projects, despite their little diffusion in comparison to other general processing units (48). In fact, there are already projects developed specifically for these processors (20).

On the other hand, projects grouped under the platform BOINC together take more than 5 PFLOPS , the largest of them **Milkyway@home**, with about 1.5 PFLOPS (70) (more then double then SETI@home).

To find more about these data they can be compared to the most powerful and expensive supercomputers currently developed, whose summarized characteristics can be seen in Figure 6.1. The Cray XT Jaguar, which currently holds the top spot in processing capacity machines for general purpose and uses 224,162 Opteron processors simultaneously, has an average yield of 1.7 PFLOPS (48). And the performance of all voluntary computing projects sums almost the same as the sum of the TOP500 computers, which covers 500 supercomputers, the most powerful in the world (36).



1	**Jaguar**, Cray XT5 6-core 2.6 GHz	DOE / OS / ORNL	USA	224162	1.76
2	**Nebulae**, Dawning TC3600 Blade, Intel X5650, NVidia Tesla C2050 GPU	National Supercomputing Centre in Shenzhen (NSCS)	China	120640	1.27
3	**Roadrunner**, IBM BladeCenter QS22/LS21 Cluster, PowerXCell 3.2 Ghz / Opteron 1.8 GHz, Voltaire Iband	DOE / NNSA / LANL	USA	122400	1.04
4	**Kraken**, Cray XT5 6-core 2.6 GHz	NSF / U of Tennessee	USA	98928	.832
5	**Jugene**, IBM Blue Gene/P Solution	Forschungszentrum Juelich	Germany	294912	.826

Figure 6.1: Currently Most Powerful Supercomputers According to June 2010 Ranking

It has to be mentioned the growing potential of these numbers, since it is estimated that by 2015 the basis of personal electronic processing devices will rise more then 2000 million (34). If we increase the base of the devices in these projects until, say 20 million (a 1 % of all devices, which is quite possible), each yielding a capacity of 250 GFLOPS (25 % of the performance of some current commercial GPUs) we will have a base of about 5 ExaFLOPS, something unimaginable to achieve with other systems within many years, under the current conditions.

Not only that, but if we compare the costs of both paradigms, result that a single server and few people will serve to maintain the distributed architecture, while the cost of one supercomputer is rarely off one million euros. Even spending a little quantity to keep the volunteers (not monetarily, but by spending on incentives or publicity) say 500 euros per 1000 users, we would be talking about something as one tenth part of the cost of a supercomputer.

In terms of sustainability or scalability, problems of high energy consumption threatens this paradigm as well as the knowledge required for mounting. However, these problems disappear with tools that facilitate system creation, extension and maintenance and with increased performance of the existing chips.

6.2 Boinc

In continuation will be explained in detail what is BOINC is and how it works, the main tool to create projects of this type at a low cost. Not all scientific communities have the knowledge to develop their own tools for distributed computing. BOINC facilitated a lot the deployment of such projects in quite economical and fast way with an open source code (47).

The first projects that used the BOINC platform were launched in 2004 by Professor David P. Anderson, UC Berkeley, one of the people who devoted a lot of his time to the voluntary computing. Currently there exist about 60 projects in execution with varying degrees of success. Even other projects that previously used their own distributed software infrastructure migrated to BOINC (as in the case of SETI@home, not as the Folding@home project whose current major support are the PlayStation 3).

Some of the most successful projects and with major processing power developed on BOINC are **Einstein@home (14), Milkyway@home (27), Rosetta@home (32), ClimatePrediction.net (10)** and **Wold Community Grid (39)** devoted to investigation in medicine, astronomy, climate, biology or math.

Within the borders of Spain there exist **GPUGrid.net** already mentioned for its innovative use of GPUs from the Research Biomedical Institute in Barcelona, and the project **Ibercivis** (22), platform driven by research centers form all around Spain with various domains of knowledge.

Simplified BOINC structure can be seen on Figure 6.2, basically consisting of two parts. On the one hand we have a server, which is responsible for creating and managing all projects, and on the other hand there is a client program distributed among the volunteers.

The projects are independent and anyone can create and add one with supervision by the developers of BOINC before they appear on the official list of projects. Apart from the BOINC infrastructure, the projects have their own server and own web support, so that the unique service provided by BOINC is own installation software and a simple Web form where the program is downloaded and where a small index of the various projects is displayed.

Once the volunteers download and install the BOINC client (8) they can join one or more projects and set the program to assign to each one a part of the resources from their computer. The fact that they can participate in several projects simultaneously is important, because many times the projects can go through periods of inactivity, or even cease to be active while the user does not even notice. This way BOINC is able to reconfigure itself to exploit the full capacity.

BOINC only gives the infrastructure, but does not validate the projects, so the user himself should ensure the validity and decide which of them, according to its opinion, deserves worthy of support. It is very important to note this. By itself the platform does not act as a filter on the home projects, which means that the users must be aware to whom they trust and provide access to their

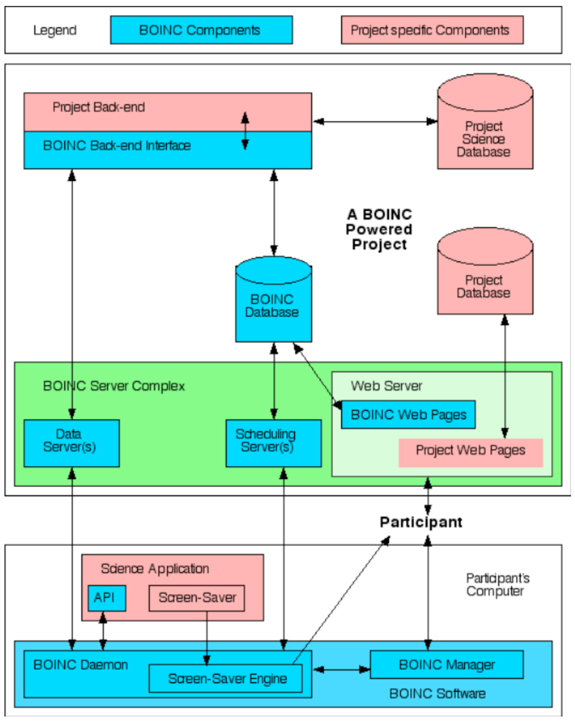


Figure 6.2: Complete Structure of the BOINC Environment

machines.

Being mounted in this way this project has a collateral advantage, because it makes the volunteers more involved in the projects. The fact that they decide to which project to join and which to avoid and to which to devote more time, requires they dedicate more interest in the evolution of these which increase their interest in the scientific subjects. Moreover BOINC allows comparison of credits among projects, to prevent a user with accumulated a lot of credits in one project be discouraged of volunteering in the others. Already has been commented that in such communities, the factor of status or rank, and the competitive factor is very important for the users.

However, the requirement of such implication and knowledge may be considered as a failure, either for security reasons, because the users can not always assure in whom to confide their machines, or the community, that loses the ability to add users who want to devote less time. As a solution for this, **account managers** have been created. They act as intermediaries and manage the projects to which an user is connecting depending on its interests, preventing it to study all the projects BOINC in case it doesn't want and also ensures their reliability and validity.

Chapter 7

Hybrid Cloud and Voluntary Computing Model?

7.1 Hybrid Architecture

As already introduced in Section 4.3.1, during the ultimate years a great interest on cloud computing has been manifested from both academic and private research centers, and numerous projects from industry and academia have been proposed. In a commercial context the most highlighted among the others are: Amazon Elastic Compute Cloud (3) IBM's Blue Cloud (23), Sun Microsystems Grid@Network.com (21), Microsoft Azure Services Platform (38). All of them support and provide an on-demand computing paradigm: a user submits his/her requests to the cloud remotely, in a distributed fashion, processes them there and receives back the results.

Most of the demanded tasks in the world of academia and science above all, require numerous computational resources which in some cases are growing dramatically faster than Moores Law as predicted by the Sun CTO Greg Papadopoulos (108), that could bring, in a close future, towards an oligarchy, a lobby or a trust of few big companies controlling the whole computing resources market.

To avoid such pessimistic but achievable scenario, there have been suggested more democratic parallel initiatives to achieve a cloud computing where the computing resources of single users are shared with the others in order to contribute to the elaboration of complex problems, at a drastically inferior cost then the commercial cloud computing. Volunteer computing has been proposed for those with a vision closer to the scientific world.

This is the objective pursued by projects with academic interest such as Reservoir (31), Nimbus/Stratus/Wispy/Kupa (112) and OpenNEbula (29) that pretend in offering open source cloud computing to the scientific would.

The cloud computing paradigm can lead from smaller scales - a unique user sharing its desktop computer with research groups public administrations to social communities, small and medium-sized companies that make their resources available to the cloud either free or paying for their use. Both these cases can be easily implementable in such scenario.

In this section will be shown how the possibility of constructing an open cloud computing architecture, thanks to the combination of the same with voluntary computing, it will be also presented an example architecture and finally some of the use cases that could bring high benefits will be discussed.

7.1.1 Synergy Between Cloud and Voluntary Computing

For years grid computing has been seen as the solution to all computational problems: a distributed computational architecture, secure and viable with an adequate performance. However it faces some disadvantages such as the incompatibility between different hardware or software, dynamic extension by adding new resources is not feasible, neither it offers opportunities for QoS and billing .

These shortcomings have been addressed with the appearance of cloud computing, by implementing service-oriented paradigms to provide a higher level interface which is more user-friendly than grid computing. Since these architectures have been primarily designed for business purposes, QoS are provided to offer better services at high prices.

The drawback of cloud computing is the lack of freedom, a free point of view to allow development of cloud services for free that way to promoting the development of new services. Another inconvenience of the existing cloud architectures is that anybody can implement its own interface and services, which are not capable of interacting with other clouds.

From other side, it is the volunteer computing that provides a distributed and open environment in which resources are shared to perform more complex tasks, yet facing the similar drawbacks of grid computing, such as the incompatibility between different resources and the inability to provide QoS and billing to get cloud architecture closer for both commercial and open source development (*Best Effort*).

It can be seen that both technologies cloud and volunteer, can complement to take the advantages of each other and to deal with the disadvantages of the other. The most important issues that should be addressed in order to achieve this objective, when implementing a new open cloud architecture are summarized within the following:

- **Resource and Service Management.** Get a mechanism to easily manage the resources and services offered by the cloud.
- **Frontend.** Create a high-level layer which abstract the complex architecture consisting of many heterogeneous resources, providing a service-oriented interface.
- **Security.** A mechanism is required to provide authentication, protection of resources, data confidentiality and integrity.
- **Interoperability among Clouds.** Possibility of different clouds to interoperate.
- **Business Models.** Offering management of QoS/SLA to achieve both commercial and open environments that would traditionally be best effort.

7.1.2 Example Architecture

One possible example architecture called **Cloud@Home** (63) is illustrated in Figure 7.1, which identifies four distinct layers: Frontend, Software Infrastructure, Software Kernel and Firmware/Hardware. There exist a distinction between two types of users:

- **End user:** users who will benefit of the cloud resources when implementing their services. What the end user see is only a storage system provided by the cloud together with the available machines in the cloud - the virtual machines.
- **User Contributor:** user who decides to share part of the cloud resources with the end users.

In continuation is briefly explained the purpose of each of the layers defined in the architecture.

7.1.2.1 Frontend

This layer is responsible for the resources and service management (enrolling, discovery, allocation, coordination, monitoring, scheduling, etc) from the global cloud systems perspective. The frontend layer provides tools for translating end-user requirements into physical resources demand, also considering QoS/SLA constraints, if specified by the user. Moreover in commercial clouds the frontend must be able to negotiate the QoS policy to be applied (SLA), monitoring their its

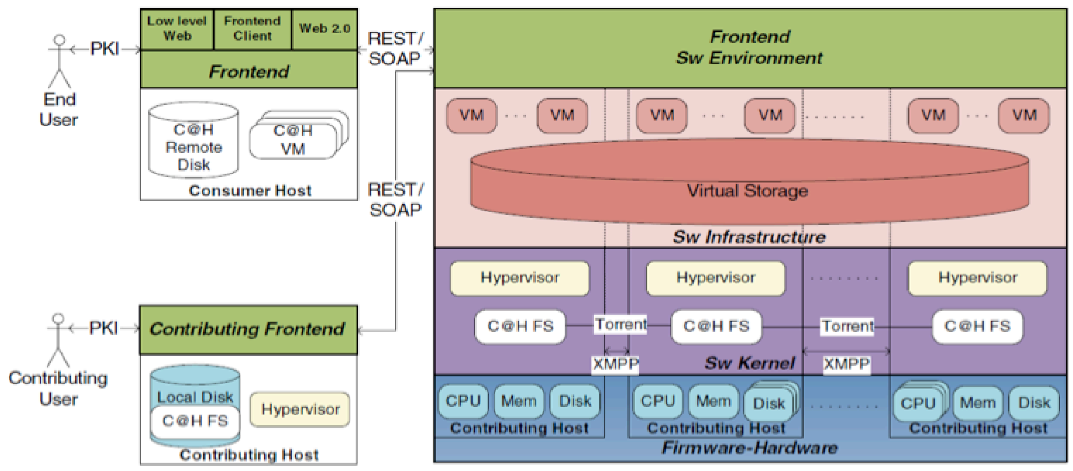


Figure 7.1: Example Architecture

fulfillment and, in case of unsatisfactory results, adapting the computing workflow to such QoS requirements.

Here comes into play the interoperability between different clouds, in case when an end user requires certain QoS and the cloud does not have sufficient resources to satisfy this QoS, this layer must be able to interoperate with other commercial or open clouds (Figure 7.2) to reserve additional resources for the end user.

The frontend layer provides also a mechanism to achieve secure user authentication and authorization mediate Public Keys Infrastructure (PKI) (128), that works better if combined with SmartCard devices.

7.1.2.2 Software Infrastructure - Virtual Layer

The virtualization of physical resources offers to end-users a homogeneous view of the available clouds services and resources, despite the significant heterogeneity of contributors, thanks to the technology of virtualization. Two basic services are provided: *computing* and *storage services.*

The computing service is achieved by the creation of virtual machines on the contributing nodes with the ability to be managed by the end users as if they were located in their own node. Thanks to the virtualization not only homogeneity of resources is achieved, but also an isolation of contributors' local resources and shared resources for the safety of both users.

The storage service implements a distributed storage system along the disk's resources that are shared by contributors who create the cloud. It is obvious that there must be a considerable redundancy due to the low availability of contributors' resources. In addition this information has to be protected to prevent unauthorized access. The end user has a vision of this service as a remote disk mounted on its own system.

7.1.2.3 Software Kernel and Hardware - Physic Layer

The last two layers visible in Figure 7.1 match the local resources of the contributing nodes, from the software (SW Kernel) to the hardware (Firmware/Hardware). This layer is divided horizontally into each of the physical devices formed by the cloud. The cloud must negotiate with the contributing user the amount of shared resources. This involves the physical layer that contains the appropriate tools to make the reservation of resources for the cloud, in addition to monitor their use in order to check compliance with the agreement and prevent that the local user runs out of resources.

For attainment of this objective it is fundamental the presence of a **hypervisor** to be installed on the contributing node for the creation of the virtual machines to be available in the upper layer. More resources to be shared, more virtual machines are created with final goal that all become exactly equal in order to facilitate the management by the frontend layer. Similarly, a small portion of local disk is reserved and assigned to the cloud which has to be protected to prevent access to information by the local user. Here the use of symmetric keys is purposed due to their speed to encrypt the information and the protection of these keys through asymmetric keys so that only authorized users can have access to the information.

Information exchange between the different nodes can be performed without the use of secure protocols such as SSL since the data is already protected, so that this communication would be established by the means of a much more efficient protocol, BitTorrent (62) that in addition benefits from the data replication over of the nodes that form the cloud.

7.1.3 Application Scenarios

Various use cases for the proposed architecture can be imagined, the most highlighted are:

- **Investigation Centers or Scientific Communities**. Volunteer computing provides the ability to create computational clouds by sharing of resources for scientific purposes as discussed in Section 6.1. However in these environments there exist compatibility problems, inability to for example make migration of some tasks from one node to another because of the resource heterogeneity seen from the end user point of view. The described architecture can be used for the same purpose with the advantage of having a homogenous vision of resources.

- **Enterprices**. For small and medium-sized enterprises the acquisition of a cloud computing service may be too costly, but using this architecture an own cloud can be easily deployed with the addition of own resources or reservation of other remote resources. Additionally the company can benefit from the sale of assets to other end users of the cloud at a time in which they are underutilized by the resources, including parts of clouds with commercial purposes - interoperability between clouds.

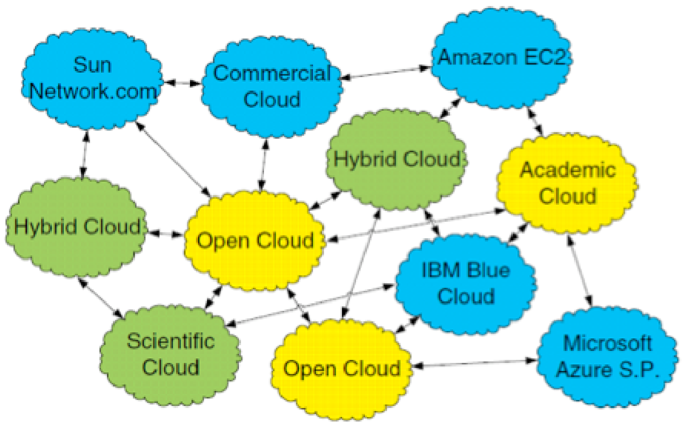


Figure 7.2: Interoperability Among Clouds

- **Individual Users**. As is the case of file sharing networks, this architecture provides a community of sharing free resources and each user makes its small contribution as a with the Long-Tail effect, where many small contributions make an enormous computational capacity. This provides the possibility for individual developers without acquisition of cloud services to launch their projects mediate the example architecture described.

Chapter 8

Conclusions and Future Research

8.1 Summary

This thesis has addressed several aspects related to the development of computing models, architectures and protocols in close relation to the models such as cloud computing, peer-to-peer and voluntary computing.

Initially starting with studious presentation the P2P model through analysis of various P2P streaming techniques, examining the associated Quality of Service (QoS) through comparing a variety of related analysis. With the presentation of research work successfully implemented in this domain, it is confirmed the there still exist good ideas that leverage the further enhancement in the domain of P2P.

Going further, the basic NAT types and utilities were investigated, addressing the NAT traversal issue in a multimedia communication environment. Summarized view of a great variety of NAT traversal techniques followed, pointing out some interesting suggestions and architectures in the world of science. The objective to establish a scenario that involves NAT and simulating the possible problems to resolve some architectural doubts was achieved. As a conclusion, could be sum up successful tests proceedings over the architecture, proper NAT and PAT translation. The existing techniques for NAT traversal work correctly when applied in P2P multimedia scenario. Furthermore NAT increases the security, especially in a vulnerable systems. Taking into consideration the QoS like delay, jitter, bandwidth and packet loss for sending a continuous stream of UDP packets through NAT, it was shown small packet loss rate which indicates that NAT traverse and network transmission performance is stable, reliable and precise. As per a current estimate, we believe NAT is not bandwidth dependent, therefore low bandwidth for example, would not directly affect the accuracy of NAT translation. Since we've been using NAT technique mediate PAT, this applies static mapping, therefore IP addresses take up very small space in the NAT address table, hence the NAT configuration saves a lot of the router's internal memory.

Cloud Computing took significant part as the main motivation in the entire investigation process. In the last few years it increasingly gained a huge momentum in the industry and the academic world and undoubtedly caused a huge turnover in the computing world. Starting form its initials, the idea, the model it follows, the various definitions and discussion, even contradictions, the functionality of the cloud computing paradigm, explaining some relevant issues that will mark its future expansion and generalization, were the carrier objectives of this thesis. All of the above and many other concerns were discussed and presented in this work.

An attempt to combine P2P and Cloud computing was predicted as a possible fusion to be treated by the research world in future. A novel architecture for P2P multimedia streaming in both client-server and P2P style was demonstrated, at the same time presenting QoS API functions implemented in a web service of the cloud streaming service provider. This approach threats the cloud model form a business point of view presenting a price model to be offered to the users of the cloud, but in the same time fills the gap between the long battle among P2P and ISPs, offering a transparent solution for both the service providers and the end users. It could be considered as an open the door for the providers of streaming services to take the advantage of such cloud paradigm in order to deploy scalable SaaS with QoS guarantees.

Apart from the architecture, it was not possible to avoid mentioning cloud security considered to be an unresolved issue and most relevant current challenge, wrapped up with the rapid movement of the cloud architectures. There are no standardized methods, neither established official protocols for security in the cloud. Therefore many security requirements were stressed for a different level of the cloud infrastructure and for each service model in particular. Ways to deal with those purposed by the experts in this area was discussed including relevant academic publications. Apparently new risks and security issues concern the customers of the cloud infrastructure, especially the SaaS users to rely completely on the cloud services, above all due to the complexity and heterogeneity of the cloud architecture, that requires different approaches from the common ones, in order to successfully deal with security.

Finally, for the same purpose of the cloud computing, the idea of voluntary computing was exposed, describing its basic points and several major projects and platforms currently under development. The potential that exists in the use of underutilized resources connected to the Internet is very high but although advances have been done, there is still a long way to go in terms of key issues such as usability and security, matching exactly with the current key concerns of cloud computing.

8.2 Outlook and Future Research

P2P peer technology, although dealing with security issues, can be advanced and implemented in many composition works and future researches. Topologies build on the top of the overlay, or cross-layer topologies can enable various functionalities for particular use, intermediate implementations or composite solutions. Finally as example of interesting solution, would be implementing the P2P multimedia streaming functionalities as an integrated topology in a Cloud Computing architecture playing around with the automatization of the QoS for increasing the users' QoE and achieve a better cost model in front of the competitive cloud providers. On the other hand P2P could easy find its place in such an architecture since as already shown on the Internet, everything that is protected and forbidden could be broken and revealed, so why not offer at the first place a model that would union this two rival technologies to interoperate together.

Related to the lower level when dealing with a protocols for application level topologies or overlays there exist several open issues in the domain of NAT. As a disputable method apart from being useful in many ways, NAT creates undesirable issue for end-to-end applications. Trying to apply some of the proposed NAT traversal solutions and examining their abilities over the proposed architecture could be one possible interesting perspective for future work. From other side, trade-off between the established parameters in the project and the QoS retrieved form the video conference topology, could be another interesting issue to observe.

In the purposed hybrid P2P and Cloud architecture was left as a future goal the implementation of the described APIs in a cloud infrastructure by extending the functionality of the Isabel project and test its behaviour. As an interesting use case to explore, could be extension of the same architecture to support videoconferencing in the cloud. Other work could focus on the possibility to unify cost policies for cloud streaming services build on different cloud infrastructures.

New ideas for investigation in the area of cloud computing would basically turn towards new model architectures and protocols as well as tuning the QoS but on a more global level. Nevertheless, most of them should consider security as a basic and inevitable part of their research whose enhancement would condition and influence a lot on the global progress of the cloud paradigm.

At the end, the cloud and voluntary computing, for the moment are following different paths but with the same objective that can be combined connecting the advantages of each other, with a perspective to strengthen the constraints that they face with nowadays. It has been shown that the convergence of cloud computing and volunteer computing can rise to new paradigms with the motivation that all the entities, from small developers to large companies can benefit from this novel computational ecosystem. Even though the example architecture related to their synergy represents an approximation or superficial final architecture, nevertheless it reflects the feasibility of this initiative.

Finally, It is not visible, neither easy predictable the future growth of cloud paradigm, but one thing is confirmed and that is the cloud dependence and tight relation with security mechanisms and protocols that are conditioning its proper deployment and secure stable, reliable and trustworthy platform for many new users to come. For the moment mixed opinions and predictions could be met coming form scientific papers and the experts related to the future advancement of cloud computing, but until the moment in which protocols are standardized and SLAs are fully guaranteed and fulfilled, it would stay as a tempting research platform for new models, and testbed for the cloud providers on their way to make it more scalable, reliable, secure and most of all available to their customers.

References

[1] Above the Clouds: A Berkeley View of Cloud Computing. White Paper. February 2009. http://berkeleyclouds.blogspot.com [June 2010]. 50

[2] Amazon CloudFront. http://aws.amazon.com/cloudfront [April 2010]. 62

[3] Amazon EC2. http://aws.amazon.com/ec2 [April 2010]. 45, 62, 78

[4] Amazon Reserved Instances. http://aws.amazon.com/ec2/reserved-instances [April 2010]. 63

[5] Amazon Simple Storage Service. http://aws.amazon.com/s3 [June 2010]. 45

[6] Amazon Web Services. http://aws.amazon.com [June 2010]. 47, 50

[7] Amazon Web Services' terms of service. http://aws.amazon.com/agreement [May 2010]. 51

[8] BOINC. http://boinc.berkeley.edu [June 2010]. 76

[9] CernVM Software Appliance. http://cernvm.cern.ch/cernvm [June 2010]. 73

[10] ClimatePrediction.net. http://climateprediction.net [June 2010]. 76

[11] Cloud computing definition. http://en.wikipedia.org/wiki/Cloud_computing/.

[12] Cloud computing definitions. http://cloudcomputing.sys-con.com/node/612375/print [June 2010]. 40

[13] Distributed.net. http://www.distributed.net [June 2010]. 71

[14] Einstein@home. http://www.einsteinathome.org [June 2010]. 76

[15] Folding@home. http://folding.stanford.edu [June 2010]. 71

[16] Folding@home PS3 FAQ. http://folding.stanford.edu/English/FAQ-PS3 [June 2010]. 74

[17] GIMPS: Finding World Record Primes Since 1996. http://www.mersenne.org [June 2010]. 71

[18] Google App Engine. http://appengine.google.com [June 2010]. 45

[19] Google App Engine's terms of service. http://code.google.com/appengine/terms.html [May 2010]. 51

[20] GPUGRID. http://www.gpugrid.net [June 2010]. 74

[21] Grid@network.com. http://www.sun.com/service/sungrid/whatsnew.jsp [June 2010]. 78

[22] Ibercivis. http://www.ibercivis.es [June 2010]. 76

[23] IBM "Blue Cloud Project". http://www.ibm.com/ibm/cloud [June 2010]. 78

[24] Isabel. http://isabel.dit.upm.es/content/view/16/34 [April 2010]. 61

[25] KMIP, the OASIS Key Management Interoperability Protocol. http://www.oasis-open.org/committees/kmip/ [May 2010]. 55

[26] Mersenne Primes: History, Theorems and Lists. http://primes.utm.edu/mersenne [June 2010]. 71

[27] MilkyWay@home. http://milkyway.cs.rpi.edu/milkyway [June 2010]. 76

[28] Musana. http://musana.com [April 2010]. 62

[29] OpenNebula Project. http://www.opennebula.org [June 2010]. 45, 79

[30] PPLive.com. 9

[31] "Reservoir Project". http://62.149.240.97 [June 2010]. 79

[32] Rosetta@home. http://boinc.bakerlab.org/rosetta [June 2010]. 76

[33] SAML, the OASIS Security Assertion Markup Language. http://saml.xml.org [May 2010]. 55

[34] SETI@home Community. http://setiathome.berkeley.edu/sah_community.php [June 2010]. 71, 75

[35] SPML, the OASIS Service Provisioning Markup Language. http://www.oasis-open.org/committees/provision [May 2010]. 55

[36] TOP500 June 2010 List. http://www.top500.org/lists/2010/06 [June 2010]. 75

[37] Vision Project. https://www.cenit-vision.org [January 2010]. 31

[38] Windows Azure Platform. www.microsoft.com/windowsazure [June 2010]. 45, 78

[39] Wold Community Grid. http://www.worldcommunitygrid.org [June 2010]. 76

[40] Wowza. http://www.wowzamedia.com/ec2.html [April 2010]. 62

[41] X.509 Certificates. http://www.itu.int/rec/T-REC-X.509/en [May 2010]. 55

[42] XACML, the OASIS eXtensible Access Control Markup Language. http://www.oasis-open.org/committees/xacml [May 2010]. 55

[43] *An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol*, 2006. 16

[44] U. Abbasi, M. Mushtaq, and T. Ahmed. Delivering scalable video coding using p2p small-world based push-pull mechanism. In *GIIS'09: Proceedings of the Second international conference on Global Information Infrastructure Symposium*, pages 94–100, Piscataway, NJ, USA, 2009. IEEE Press. 12

[45] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPs and P2P users cooperate for improved performance? *SIGCOMM Comput. Commun. Rev.*, 37(3):29–40, 2007. 62

[46] M. Alhaisoni, M. Ghanbari, and A. Liotta. Improving qos in p2p video streaming. pages 98 –103, july 2009. 11, 14, 15

[47] D. P. Anderson. Boinc: A system for public-resource computing and storage. In *GRID '04: Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, pages 4–10, Washington, DC, USA, 2004. IEEE Computer Society. 76

[48] D. P. Anderson. Volunteer computing: the ultimate cloud. *Crossroads*, 16(3):7–10, 2010. 74, 75

[49] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer. SETI@home: an experiment in public-resource computing. *Commun. ACM*, 45(11):56–61, 2002. 71

[50] A. K. Andreas Muller and G. Carle. ANTS - A framework for knowledge based NAT traversal - draft version. `http://131.159.15.247/pubs/globecom09-draft.pdf` [January 2010]. 26, 27

[51] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50–58, 2010. 41, 47

[52] P. Baccichet, J. Noh, E. Setton, and B. Girod. Content-aware p2p video streaming with low latency. In *IEEE Int. Conference on Multimedia and Expo, ICME*, 2007. 12

[53] S. A. Baset and H. Schulzrinne. An analysis of the Skype Peer-to-Peer internet telephony protocol. Technical report, Department of Computer Science, Columbia University, 2004. 31

[54] M. S. Blumenthal. Hide and seek in the cloud. *IEEE Security and Privacy*, 8:57–58, 2010. 50, 51

[55] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.*, 25(6):599–616, 2009. 44

[56] H. Casanova, A. Legrand, and M. Quinson. Simgrid: A generic framework for large-scale distributed experiments. In *UKSIM '08: Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, pages 126–131, Washington, DC, USA, 2008. IEEE Computer Society. 73

[57] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *SOSP '03: Proceedings of the nineteenth ACM symposium on*

Operating systems principles, pages 298–313, New York, NY, USA, 2003. ACM. 8

[58] K.-T. Chen, C.-Y. Huang, P. Huang, and C.-L. Lei. Quantifying skype user satisfaction. *SIGCOMM Comput. Commun. Rev.*, 36(4):399–410, 2006. 18

[59] M. Christodorescu, R. Sailer, D. L. Schales, D. Sgandurra, and D. Zamboni. Cloud security is not (just) virtualization security: a short paper. In *CCSW '09: Proceedings of the 2009 ACM workshop on Cloud computing security*, pages 97–102, New York, NY, USA, 2009. ACM. 59

[60] Y.-h. Chu, S. G. Rao, and H. Zhang. A case for end system multicast (keynote address). In *SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 1–12, New York, NY, USA, 2000. ACM. 7

[61] Cisco. How NAT works. `http://www.cisco.com/en/US/tech/tk648/tk361/technologies_tech_note09186a0080094831.shtml` [January 2010]. 21

[62] B. Cohen. "The BitTorrent Protocol Specification". `http://www.bittorrent.org/beps/bep0003.html` [June 2010]. 5, 9, 61, 82

[63] V. D. Cunsolo, S. Distefano, A. Puliafito, and M. Scarpa. Cloud@home: bridging the gap between volunteer and cloud computing. In *ICIC'09: Proceedings of the 5th international conference on Emerging intelligent computing technology and applications*, pages 423–432, Berlin, Heidelberg, 2009. Springer-Verlag. 80

[64] Davis, Read, Scott, and Cordell. Traversal on non-protocol aware firewalls and NATs. Internet Engineering Task Force Internet-draft. Work in progress, expires April 2002. 28

[65] M. D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, and A. Vakali. Cloud computing: Distributed internet computing for it and scientific research. *IEEE Internet Computing*, 13:10–13, 2009. 59

[66] O. Dmytrienko, B. Bilodid, and M. Ternovoy. Comparative analysis of the voip software. pages 398 –399, feb. 2009. 17

[67] B. Donassolo, H. Casanova, A. Legrand, and P. Velho. Fast and scalable simulation of volunteer computing systems using simgrid. In *Workshop on Large-Scale System and Application Performance (LSAP)*, 2010. 73

[68] ENISA. Cloud Computing Security Risk Assessment. White Paper. November 2009. http://www.enisa.europa.eu/act/rm/files/\deliverables/cloud-computing-risk-assessment. 59

[69] J. L. Eppinger. TCP Connections for P2P Apps: A Software Approach to Solving the NAT Problem. *Carnegie Mellon University*, January 2005. 31

[70] T. Estrada, M. Taufer, and D. P. Anderson. Performance prediction and analysis of boinc projects: An empirical study with emboinc. *J. Grid Comput.*, 7(4):537–554, 2009. 74

[71] B. Ford, P. Srisuresh, and D. Kegel. Peer-to-peer communication across network address translators. In *ATEC '05: Proceedings of the annual conference on USENIX Annual Technical Conference*, pages 13–13, Berkeley, CA, USA, 2005. USENIX Association. 30

[72] M. Fouquet, H. Niedermayer, and G. Carle. Cloud computing for the masses. In *U-NET '09: Proc. of the 1st ACM workshop on User-provided networking: challenges and opportunities*, pages 31–36. ACM, 2009. 60

[73] S. Fraser, R. Biddle, S. Jordan, K. Keahey, B. Marcus, E. M. Maximilien, and D. Thomas. Cloud computing beyond objects: seeding the cloud. In *OOPSLA '09: Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 847–850, New York, NY, USA, 2009. ACM. 46, 48

[74] S. L. Garfin. VoIP and Skype Security. 17

[75] X. Gou and W. Jin. Multi-agent system for multimedia communications traversing nat/firewall in next generation networks. In *Communication Networks and Services Research, 2004. Proceedings. Second Annual Conference on*, pages 99–104, May 2004. 20, 29

[76] C. U. C. D. Group. Cloud Computing Use Cases. White Paper. 2nd February 2010. http://groups.google.com/group/cloud-computing-use-cases. 53, 54

[77] S. Guha. Nutss: a SIP-based approach to UDP and TCP network connectivity. In *In SIGCOMM 2004 Workshops*, pages 43–48. ACM Press, 2004. 30

[78] S. Guha, N. Daswani, and R. Jain. An experimental study of the skype peer-to-peer voip system. In *IPTPS'06: The 5th International Workshop on Peer-to-Peer Systems*. Microsoft Research. 16

[79] S. Guha and P. Francis. Characterization and measurement of TCP traversal through NATs and firewalls, 2005. 30

[80] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava. Promise: Peer-to-peer media streaming using collectcast, 2003. 15

[81] M. Holdrege and P. Srisuresh. Protocol complications with the IP network address translator. RFC 3027. Network Working Group. January 2001. `http://www.ietf.org/rfc/rfc3027.txt` [January 2010]. 23, 31

[82] C. Huitema. Teredo: Tunneling IPv6 over UDP through network address translations NATs. RFC 4380. Network Working Group. February 2006. `http://tools.ietf.org/html/rfc4380`. 31

[83] K. Hwang, S. Kulkareni, and Y. Hu. Cloud security with virtualized defense and reputation-based trust mangement. *Dependable, Autonomic and Secure Computing, IEEE International Symposium on*, 0:717–722, 2009. 59

[84] I. S. in Storage Working Group. IEEE P1619. `https://siswg.net` [May 2010]. 55

[85] W. Itani, A. Kayssi, and A. Chehab. Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures. *Dependable, Autonomic and Secure Computing, IEEE International Symposium on*, 0:711–716, 2009. 59

[86] L. J, C. PA, and Z. C. Mutualcast: an efficient mechanism for content distribution in a P2P network. In *Acm Sigcomm Asia Workshop*, Beijing, China. Apr. 10-12, 2005. 7

[87] S. Jang and M. Kent. Multiple subscriber videoconferencing system. United States Patent Application Publication, September 2005. 28

[88] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr. Overcast: reliable multicasting with on overlay network. In *OSDI'00: Proceedings of the 4th conference on Symposium on Operating System Design & Implementation*, pages 14–14, Berkeley, CA, USA, 2000. USENIX Association. 7

[89] B. Javadi, D. Kondo, J.-M. Vincent, and D. Anderson. Mining for statistical models of availability in large-scale distributed systems: An empirical study of SETI@home. In *Modeling, Analysis Simulation of Computer and Telecommunication Systems, 2009. MASCOTS '09. IEEE International Symposium on*, pages 1 –10, 21-23 2009. 73

[90] M. Jensen, J. Schwenk, N. Gruschka, and L. L. Iacono. On technical security issues in cloud computing. *Cloud Computing, IEEE International Conference on*, 0:109–116, 2009. 59

[91] B. R. Kandukuri, R. P. V., and A. Rakshit. Cloud security issues. In *SCC '09: Proceedings of the 2009 IEEE International Conference on Services Computing*, pages 517–520, Washington, DC, USA, 2009. IEEE Computer Society. 59

[92] L. M. Kaufman. Data security in the world of cloud computing. *IEEE Security and Privacy*, 7(4):61–64, 2009. 59

[93] L. M. Kaufman. Can a trusted environment provide security? *IEEE Security and Privacy*, 8:50–52, 2010. 59

[94] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat. Bullet: high bandwidth data dissemination using an overlay mesh. In *Proceedings of the 19th ACM Symposium on Operating System Principles (SOSP)*, 2003. Dept. of Comput. Sci., Duke Univ., Durham, NC, USA. 8

[95] Lakhani, K. R., Wolf, and R. G. Why hackers do what they do: Understanding motivation and effort in free/open source software projects. *SSRN Electronic Journal (2003)*, pages 1–27, 2003. 71

[96] S. Larson, C. Snow, M. Shirts, and V. Pande. Folding@home and genome@home: Using distributed computing to tackle previously intractable problems in computational biology. 2009. 71

[97] J. Lennox and H. Schulzrinne. A protocol for reliable decentralized conferencing. In *NOSSDAV '03: Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*, pages 72–81, New York, NY, USA, 2003. ACM. 17

[98] H. Li, Y. Dai, L. Tian, and H. Yang. Identity-based authentication for cloud computing. In *CloudCom '09: Proceedings of the 1st International Conference on Cloud Computing*, pages 157–166, Berlin, Heidelberg, 2009. Springer-Verlag. 59

[99] J. Li. On peer-to-peer (p2p) content delivery. volume 1, pages 45 –63, March 2008. 11, 17

[100] G. Lisha and L. Junzhou. Performance analysis of a p2p-based voip software. In *AICT-ICIW '06: Proceedings of the Advanced Int'l Conference*

on Telecommunications and Int'l Conference on Internet and Web Applications and Services, page 11, Washington, DC, USA, 2006. IEEE Computer Society. 17

[101] Y. Liu, Y. Guo, and C. Liang. A survey on peer-to-peer video streaming systems. volume 1, pages 18–28, March 2008. 8

[102] Y. Liu and J. Pan. The impact of NAT on BitTorrent-like P2P systems. In *Peer-to-Peer Computing, 2009. P2P '09. IEEE Ninth International Conference on*, pages 242–251, Sept. 2009. 30

[103] X. Lou, K. Hwang, and G. Xie. Quality of service in peer-to-peer iptv networks. pages 9 –17, dec. 2009. 13

[104] N. Magharei and R. Rejaie. Understanding mesh-based peer-to-peer streaming. In *NOSSDAV '06: Proceedings of the 2006 international workshop on Network and operating systems support for digital audio and video*, pages 1–6, New York, NY, USA, 2006. ACM. 10

[105] N. Magharei and R. Rejaie. Mesh or multiple-tree: A comparative study of live p2p streaming approaches. In *INFOCOM 2007, Anchorage, Alaska,6-12*, pages 1424–1432, 2007. 8, 9

[106] Y. Mansour and B. Patt-Shamir. Jitter control in qos networks. *IEEE/ACM Trans. Netw.*, 9(4):492–502, 2001. 13

[107] J. Mäntylä. Scalability of Peer-to-Peer Systems. Seminar on Internetworking. Helsinki University of Technology, 2005. 11

[108] R. Martin. "The Red Shift Theory". `http://www.informationweek.com` [June 2010]. 78

[109] A. B. Mohammed, J. Altmann, and J. Hwang. Cloud computing value chains: Understanding businesses and value creation in the cloud. In *Economic Models and Algorithms for Distributed Systems*, pages 187–208. ACM, 2010. 59

[110] A. K. Muller and G. Carle. Behavior and classification of NAT devices and implication for NAT traversal - draft version. `http://nattest.net.in.tum.de/pubs/2008IEEE-DRAFT-VERSION.pdf` [January 2010]. 23

[111] M. Mushtaq and T. Ahmed. Qos provisioning for video streaming over sp-driven p2p networks using admission control. pages 1 –2, jan. 2009. 11

[112] U. of Chicago-University of Florida-Purdue University-Masaryk University. Nimbus-stratus-wispy-kupa projects. http://workspace.globus.org/clouds/nimbus.html/,
http://www.acis.ufl.edu/vws/,
http://www.rcac.purdue.edu/teragrid/resources/#wispy,
http://meta.cesnet.cz/cms/opencms/en/docs/clouds [June 2010]. 79

[113] A. C. Oliveira, L. Sampaio, S. F. Fernandes, and F. Brasileiro. Adaptive sabotage-tolerant scheduling for peer-to-peer grids. *Dependable Computing, Latin-American Symposium on*, 0:25–32, 2009. 73

[114] D. Raz, J. Schoenwaelder, and B. Sugla. An SNMP application level gateway for payload address translation. RFC 2962. Network Working Group. October 2000. http://tools.ietf.org/html/rfc2962. 30

[115] J. P. Robinson. Optimum golomb rulers. *IEEE Trans. Comput.*, 28(12):943–944, 1979. 71

[116] J. Rosenberg. Session traversal utilities for NAT (STUN). RFC 5389. Internet Engineering Task Force. October 2008. http://www.ietf.org/rfc/rfc5389.txt [January 2010]. 24

[117] J. Rosenberg. STUN - Simple traversal of user datagram protocol (UDP)-through network address translators (NATs). RFC 3489. Internet Engineering Task Force. March 2003. http://www.ietf.org/rfc/rfc3489.txt. 17, 21

[118] J. Rosenberg. Traversal using relays around NAT (TURN): Relay extensions to session traversal utilities for NAT (STUN), draft-ietf-behave-turn-16. BEHAVE WG. July 2009. http://tools.ietf.org/html/draft-ietf-behave-turn-16 [January 2010]. 25

[119] J. Rosenberg. Interactive Connectivity Establishment. *IETF Journal*, Volume 2(Issue 3), November 2006. 26

[120] J. Rosenberg, J.Weinberger, and H.Schulzrinne. NAT friendly SIP , Internet Engineering Task Force Internet-draft. Work in progress, expires February 2002. http://www.softarmor.com/wgdb/docs/draft-rosenberg-sip-entfw-02.txt. 29

[121] L. F. G. Sarmenta. Bayanihan: Web-based volunteer computing using java. In *WWCA '98: Proceedings of the Second International Conference on Worldwide Computing and Its Applications*, pages 444–461, London, UK, 1998. Springer-Verlag. 73

[122] B. Segal, P. Buncic, D. G. Quintas, D. L. Gonzalez, A. Harutyunyan, J. Rantala, and D. Weir. Building a volunteer cloud. 2009. 73

[123] S. Sen, P. Sollee, and S. March. Midcom-unaware NAT/Firewall traversal. Midcom Working Group Internet-draft. Work in progress, expires March 2002. http://potaroo.net/ietf/all-ids/draft-sen-midcom-fw-nat-00.txt. 29

[124] E. Setton, J. Noh, and B. Girod. Low latency video streaming over peer-to-peer networks. *Multimedia and Expo, IEEE International Conference on*, 0:569–572, 2006. 12

[125] Z. Shen and R. Zimmermann. ISP-friendly peer selection in P2P networks. In *MM '09: Proc. of the seventeen ACM international conference on Multimedia*, pages 869–872. ACM, 2009. 62

[126] R. Swale, P. Mart, P. Sijben, S. Brim, and M. Shore. Best current practices for NAT traversal for client-server SIP. SIPPING Working Group Internet-draft. expires March 2009. http://tools.ietf.org/html/draft-ietf-sipping-nat-scenarios-09. 30

[127] R. Swale, P. Mart, P. Sijben, S. Brim, and M. Shore. Middlebox communications (midcom) protocol requirements. RFC 3304. Network Working Group. August 2002. http://www.ietf.org/rfc/rfc3304.txt. 30

[128] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Internet X.509 Public Key Infrastructure. RFC 3820. 2004. http://www.ietf.org/rfc/rfc3489.txt [June 2010]. 81

[129] UPnP-Forum. Internet gateway device (IGD) standardized device control protocol. November 2001. http://www.upnp.org [January 2010]. 25

[130] J. Voas and J. Zhang. Cloud computing: New wine or just a new bottle? *IT Professional*, 11:15–17, 2009. 43

[131] Wang, L. von Laszewski, G. Kunze, Marcel, Tao, and Jie. Cloud computing: A perspective study. In *Proceedings of the Grid Computing Environments (GCE) workshop*, New York, NY, USA, 2008. ACM.

[132] C. Wang, Q. Wang, K. Ren, and W. Lou. Ensuring data storage security in cloud computing. In *IWQoS '09: Proc. of the 17th International workshop on Quality of Service*, pages 1–9. IEEE, 2009. 51, 58

[133] J. Wang, C. Huang, and J. Li. On ISP-friendly rate allocation for peer-assisted VoD. In *MM '08: Proc. of the 16th ACM international conference on Multimedia*, pages 279–288. ACM, 2008. 62

[134] J. Wang, Y. Zhao, S. Jiang, and J. Le. Providing privacy preserving in cloud computing. In *ICTM '09: Proceedings of the International Conference on Test and Measurement*, pages 213–216. IEEE, 2009. 59

[135] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou. Enabling public verifiability and data dynamics for storage security. Cryptology ePrint Archive, Report 2009/281, 2009. http://eprint.iacr.org. 58

[136] X. Wang and Q. Deng. *VIP: A P2P Communication Platform for NAT Traversal*, volume Volume 3758/2005, pages 1001–1011. Springer Berlin / Heidelberg, October 2005. 27

[137] X. Wu. Sip on an overlay network. 20

[138] D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava. On peer-to-peer media streaming. pages 363 – 371, 2002. 11, 50, 57

[139] J.-S. Xu, R.-C. Huang, W.-M. Huang, and G. Yang. Secure document service for cloud computing. In *CloudCom '09: Proceedings of the 1st International Conference on Cloud Computing*, pages 541–546, Berlin, Heidelberg, 2009. Springer-Verlag. 56, 58

[140] S. Yang, H. Jin, and X. Tu. Tobab: A trend-oriented bandwidth adaptive buffering in peer-to-peer streaming system. 13

[141] G. Zheng, Z. Xie, B. Wang, and G. He. A new tunnel scheme for multimedia communications traversing nat/firewall in ngn. In *Parallel and Distributed Computing, Applications and Technologies, 2005. PDCAT 2005. Sixth International Conference on*, pages 171–173, Dec. 2005. 28

[142] R. Zhou and K. Hwang. Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing. *IEEE Trans. Parallel Distrib. Syst.*, 18(4):460–473, 2007. 14

[143] H. Zhu. Towards a theory of universally composable cloud computing. In *CloudCom '09: Proceedings of the 1st International Conference on Cloud Computing*, pages 638–643, Berlin, Heidelberg, 2009. Springer-Verlag. 58