# Designing an unified P2P protocol for file sharing and internet telephony

Hema Swetha Koppula
4th Year B.Tech.
hema.swetha@gmail.com

Tathagata Das
4th Year B.Tech.
tathagata.das1565@gmail.com

Niloy Ganguly
Assistant Professor
niloy@cse.iitkgp.ernet.in

Department of Computer Science Engineering
Indian Institute of Technology, Kharagpur
Kharagpur, West Bengal - 721 302, India

*Abstract*— **This paper is a positional paper where we propose steps for developing a unified protocol to support file transfer and internet telephony simultaneously over a P2P network. In order to develop a comprehensive design a thorough study of the KaZaA and Skype protocols, the two most popular softwares in the file sharing and internet telephony, is required. We therefore undertake a detailed literature survey to identify the unique and interesting features of KaZaA and Skype. Since these softwares are not open source, we need to perform reverse-engineering to experimentally identify quantitative and qualitative differences in order to obtain a better insight of their protocols. Finally we propose the unified architecture and provide rationale for our design.**

## I. Introduction

The internet has been the breeding ground for innumerable innovative ideas. Peer-to-peer software paradigm is another idea that has been born from the vast network and infinite resources present in the internet. As software requirements are getting larger, speed along with scalability and robustness is becoming a necessity; distributed systems like peer-to-peer softwares are truly becoming the software architecture of the future.

The most influential peer-to-peer softwares till now have been file-sharing softwares like Napster, Gnutella, and KaZaA. Only very recently, the concept of internet telephony using peer-to-peer networks has come up and P2P based Voice-over-IP (VoIP) softwares like Skype are gaining much popularity. These two types of software individually have scaled to millions of users, proving their quality. Similar to these, more such P2P oriented services will keep coming up in future. The protocol used by each of them is different although on analysis it is found that there is great deal of similarities between them. In this state of affairs, there is a need for a common distributed P2P architecture on top of which all such services can be designed, instead of each of them being designed separately from scratch.

Therefore, our objective is to design a new protocol which can provide the basic P2P networking architecture to support both file sharing and internet telephony.

The paper is organized in 3 main parts. Section 2 deals with our line of attack: how we are approaching the problem and what we need to do to achieve our objective. Section 3 deals with the current status of our work and the results that we have already produced. Section 4 deals with what we plan to do in the future.

## II. Proposed Line of work

The two most popular P2P networking softwares based on file sharing and internet telephony are KaZaA and Skype respectively. Our aim is to study the individual features of the protocols used by them and identify the differences between them. The identification is carried out by theoretical literature survey as well experimentation. Based on the analysis of the protocols used in these two softwares and their performances in terms of few metrics, defined later on, we plan to design the protocol which would allow interoperability of both file sharing and internet telephony, along with providing good Quality of Service (QoS).

### A. Similarities between KaZaA and Skype

The protocols used by KaZaA and Skype are largely similar. They both used a two-tiered peer-to-peer network, where all the peers a divided into 2 classes: supernode (SN) and ordinary node (ON) [2, 3, 4, 6, 7]. The super nodes provide higher bandwidth and are more stable clients. They interconnect among each other strongly to produce a backbone for the P2P network. The ordinary nodes connect to one of the supernodes to get connected to the network. Data transfers take place directly between the peers. This basic structure being the same, there are some important differences between KaZaA and Skype.

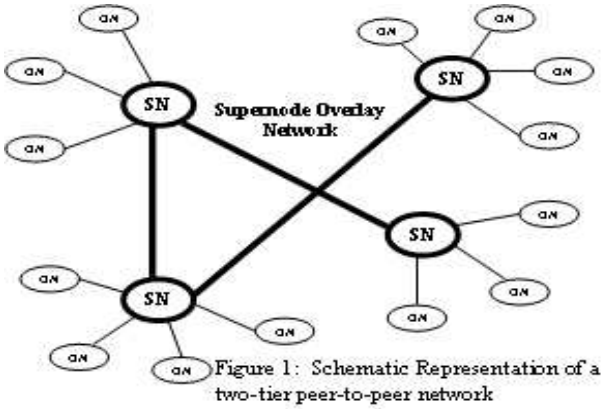

Figure 1: Schematic Representation of a two-tier peer-to-peer network

## B. Differences: Identification and reasons

Identification of the differences is necessary to understand the mutual compatibilities of different properties of the protocols. Here we state the differences based on the theoretical survey. We also figure out the reasons behind the existence of such differences in the protocols.

### B.1 Centralized login system:

Skype, being a login-based messenger service, has to provide secure login system for millions of users in the P2P network. For this purpose, it has a centralized login server, which a client has to connect to, in order to login into the Skype network [6]. The centralized login server makes it easier to maintain all the usernames, passwords and buddy lists, along with functionalities like authentication, registrations of new users, etc.

On the other hand, KaZaA does not need to maintain user accounts, so it does not have any centralized server related to its P2P network [2, 3]. It is a pure multi-tier P2P network.

### B.2 Multiple routes for data transfer

In case of Skype, to provide good quality real-time voice data, transmission delay, latency and jitter need to be minimized. For this purpose, the participating Skype clients maintain multiple possible routes, and use them alternately or in parallel for transmission. It may select the routes based on parameters such as high bandwidth, less delay, less congestion, etc and accordingly optimize the data transmission.

KaZaA, being only for file transfers, requires QoS of different criteria - data integrity. The file transfers are through HTTP protocol and generally can take place over hours, days and even weeks. This period depends on file size, available bandwidth and also participation level of the client. More importantly, immediate response is not required by the user, which is why KaZaA does not optimize the speed and the transfers usually continue in the background.

### B.3 Relay nodes for data transfer

One of the main reasons behind the popularity of Skype is its ability to operate through a large number of types of internet connection, be it behind a NAT (Network Address Translation), or behind a UDP-restricted firewall. This capability of Skype comes from its ability to use a third peer to setup a connection between two peers that want to initiate a call, but are unable to connect to each other directly. These redirecting peers are called relay nodes, and they are usually nodes having a public IP address and higher bandwidth. The relay node simply initiates connection to the 2 participating peers, and sends the real time voice data from one to the other.

KaZaA, being a much earlier product, does not have the facility of relay nodes, hence cannot operate through NATs and firewalls, without additional configuration.

### B.4 Search infrastructure

The protocol used for searching and data transfer is dependent on the type of connection that a Skype client has (i.e. public IP address or behind a NAT or behind firewall). The clients behind UDP restricted firewall cannot contact other nodes directly. So it has no option but make the SN it is connected to, to do all the processing regarding the searching for another user. But in other cases (public IP or behind NAT) the ON has the capability to do take up part of the search workload by contacting other nodes directly. This is what is being done in Skype [6]. The latter set of peers, with the help of the SNs do most of the searching themselves. This distributes the entire workload of the SN over the ON network, enabling the super peers to devote more processing power and bandwidth to other work items like messaging, call initiation, file transfer connection setup, etc. By comparison KaZaA uses the FastTrack 2 Protocol as the search architecture. Here, the SNs maintain all the list of files shared by the ONs connected to each of them. When an ON sends a search request to its SN, the SN does all the searching and compiling of the search results for the ON. So the SN always carries the complete search workload.

## C. Testing the differences

In order to extend our knowledge acquired through literature survey, detailed experimentation to understand Skype and KaZaA protocols is undertaken. Since both KaZaA and Skype are proprietary by nature, reverse engineering is the only way to understand these protocols. For this purpose we are setting up a test

system of computers on which KaZaA and Skype are run. All the traffic passing through the test system is intercepted and captured using a packet-sniffing and protocol analyzer softwares. From the captured packets we can analyze the different properties of the protocol.

C.1 Centralized login system

The exact details of how the login server interacts with the nodes for authentication will enable us to understand the pros and cons of using a centralized login system. For this purpose we propose to analyze the connections established and the TCP and UDP messages passed between the Skype client, its supernode and login server, while logging in to the Skype network. This will be performed many times and in different scenarios like successful login, unsuccessful login, etc, to get a fair idea of the interaction among the ON, SN and the Login Server at time of logging in.

C.2 Multiple routes for data transfer

The questions to which answers are not available in literature are - whether the multiple routes maintained are application-level routes (multiple relay nodes) or IP-level routes (multiple TCP connections over same route) used? In either of the cases, the number of routes and the conditions under which they are used need to be studied. We need to decipher the multiple-route infrastructure to understand properties like bandwidth usage, protocol overhead, etc. Using a node with public IP and / or NAT on both sides, the TCP connections that are being established at call initiation and maintained during the period of the call, need to be tested out to answer these questions.

C.3 Relay nodes for data transfer

The use of relay nodes being a good plus point, we need to investigate this feature to find out exactly how the connections for VoIP and Skype file transfers are initiated using a relay node. Using a test system of two computers, we can analyze the interaction between the relay node and the super node when a call or a file transfer is being initiated using the relay node. Also, Skype has the facility of transferring files. This is visibly different from the file sharing and transfer facility of KaZaA, because in Skype, we are "pushing" the file-transfer from one user to another. In case of KaZaA, the file is shared by a user, and another user "pulls" the file for the transfer to take place. The details of the protocol in both "push" and "pull" system should be understood well enough to design our own system.

C.4 Search infrastructure

1. By analyzing the message sequence that is generated while searching in Skype client, we can find the repetitive nature of the IP addresses that are being used for searching. This has to be done large number of times so that we can be sure of the statistics. Various scenarios such as different user logins, different SN connections, etc will be tested out. This will enable us to understand whether there are more than two tiers in the network and whether there is any further substructure that is dedicated for searching.
2. Using test system having a Skype client with public IP address such that it becomes a supernode, we can connect a normal node to this test supernode and then test out the ON-SN interactions and SN-SN interaction from the supernode's point of view. This will be performed for the different scenarios (public IP address/ NAT / firewall-ed normal node) that affect the search protocol.

## D. Merging the protocols

For designing a multi-service unified protocol, we need to decide on the architecture of the peer-to-peer network. Centralized login server based systems are generally more efficient than pure peer-to-peer systems, but they lack the scalability and robustness of the P2P systems. In the latter, as more and more nodes join the system, the total resource of the system increase, which is why they can easily scale up to millions of nodes. Each node adds to resource and gets the service in return. For this purpose, our aim is to design a purely multi-tier peer-to-peer system, which does not have any central component like a Login Server, and yet has enough robustness and reliability required by the system. Our objective is to store all the user account information in a distributed, yet robust manner such that searches on the distributed system give a certain degree of guarantee to find results. We will simulate the protocol to fine tune its parameters, for producing best performance in terms of robustness, reliability and efficiency.

Regarding our unified protocol, we are proposing the following the salient features:

1. Both the services, internet telephony and file sharing, will be user-account based. That is, the users have to login with their unique usernames and passwords, to gain access to the services provided by the system. During logging into the network, the ON will connect to the SN first. The SN it connects to will provide the address of the login server. This is done so that the system is flexible enough to shift login requests from one login server to another for both load-balancing and fault tolerance of servers.
2. By using supernode based indexing, the overlay network will maintain not only the unique usernames, passwords but it will also store the con-

nection state of the each user (whether connected or not) and if connected then what is its address and its supernode's address. The address of an ON's supernode needs to be known as the supernode acts as the gateway for contacting the ON. Any request from an ON will be processed by the overlay network and accordingly the reply will be sent back to it.

3. If the user has been successfully authenticated, then the ON will upload the shared file list (file information and hash values) to the supernode it is connected to.
4. For unified user and file searching, the client, with the help of its supernode, will contact other nodes in the network and search for the required data itself. This distributes the search load over all the nodes (both SNs and ONs) in the network, thus increasing the scalability of the system. In the present scenario, since the most widely used dialup lines are becoming more stable and gaining bandwidth, distributing the load over the ONs connected through dialup lines is a feasible option.
5. For real time voice transfer, if the direct connection is possible between the two participating nodes, then direct connection will be made. Other wise relay nodes will be used, and multiple parallel routes will be maintained for transmitting real-time data. In case of file transfer, multiple route protocol will be used to facilitate multi-source file transfer.

## E. Performance Evaluation

The proposed protocol must be evaluated in terms of certain performance metrics to prove its efficiency, reliability and robustness. These performance metrics can be categorized into 2 parts: general P2P network based metrics, and specific service based metrics. The following are the metrics on which we plan to test our protocol on.

1. Common P2P metrics: Probability of search success, Hop count, Avg. msg count per node, No. of nodes visited for searching, Peak msg count [9].
2. Service specific metrics:
   - VoIP: Latency, Jitter and Packet loss
   - File transfer: Data Integrity, Speed/Bandwidth, Reliability

# III. Ongoing work and results

Released in 2001, KaZaA has already been studied extensively and its protocol has been hacked to develop clone versions like KaZaA Lite, etc. In contrast, Skype was released in 2003 and because of high level of encryption its protocol is still a mystery. So we started our initial experiments with Skype. We are only at a very preliminary stage of our experiments and the set of experiments we are conducting currently deals with the interaction between the ON, the SN and the Login Server at the time of logging in. The details of our current experiments are as given below.

## A. Experimental Setup

Our experimental setup consists of a computers having connection to the internet through a NAT. It is running on Microsoft Windows XP with Skype (version 2.5.0.126) installed. We are sniffing the transferred packets by using well known packet-sniffing softwares - Ethereal (v 0.99) and EtherDetect (v1.2).

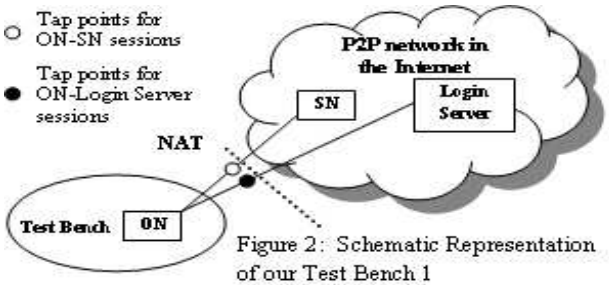

Figure 2: Schematic Representation of our Test Bench 1

## B. Experiments

Using the Ethereal and EtherDetect, we analyzed the packets that were being sent from and received by our test machine when we try to login into the Skype client. Since the level of encryption used in Skype is very high, it is hard to understand exactly what information is being transferred. Thus, just from the UDP and TCP connections, and the length, timings and durations of their conversations, we are trying to get an insight into the interaction of the ONs, SNs and the login server. The following cases were tested out with respect to logging in of a client:

1. Logging in for the first time after installing the Skype client.
2. Logging in using a correct username and password.
3. Logging in using incorrect username or incorrect password.

Each case was tested out repeatedly to be sure of the results as given below.

## C. Results and Inferences

We have got the following results from our experiments:

1. The pattern of message transfers that take place between the Skype client, the supernode and the login server, in the case where the Skype client is behind a NAT:
   - Since the login server needs to know the SN to which a client is connected, the Skype client first connects to the SN, before trying to authenticate using the login server. The Skype client (ON)

begins by trying to contact a number of SNs that it has in its 'host cache'. Then the client makes a TCP connection with the first SN that replies by UDP.

- Through the TCP connection, the client receives the address of the login server it should contact, and accordingly contacts it for authentication. This is done in this way because there are multiple login servers and SN gives the client which one to connect to. This makes the system not only more flexible, but also fault tolerant (in cases, where some login server goes down).

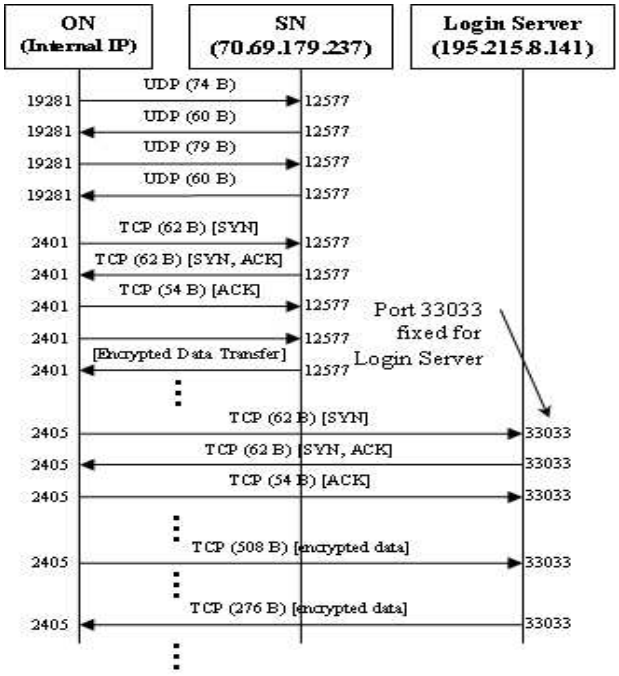


Figure 3: Schematic Representation of the sequence of messages exchanged with SN and Login Server by the ON while logging into the Skype network.

- A packet of around 508 bytes is sent to the login server as the authentication request. If successfully authenticated, the server replies with a packet of around 276 bytes, otherwise it replies with a smaller packet of 60 to 70 bytes.
- As long as the TCP connection between the ON and the SN is maintained, a regular pattern of message transfers was observed, with a periodicity of 1 to 2 minutes. Once the Skype client connects to the Skype network, the SN-ON connection is maintained even if the user has logged out of the Skype software on the ON.

2. When the Skype client is just installed, it does not have any list of SNs to which it will attempt to connect. It first tries to connect to a set of 7 hard-coded SNs, known as bootstrap SNs , from which it receives and then later maintains a list of 200 SNs for it to connect to the network.

## IV. Future work and Conclusion

As our future steps, we are planning to develop a better experimental setup comprising of two systems having public IP address, along with another system behind a NAT. We will be continuing our tests to find out different properties of the protocol with respect to search, data transfer, etc. Parallel to this, we will be developing a simulated network of our own, in which we will be testing out our unified protocol. This will be done in step by step manner, starting from a very simple peer-to-peer system, to which the features of the protocols will be added progressively.

We believe the concept of unified protocol in P2P network is an important emerging area of research and deemed attention. Therefore, we decided to report our work although it is still in a preliminary stage. This paper outlines the idea and motivation behind our work, along with the different steps that will lead us to our goal. In our current state, we have completed the literature survey and based on our findings we have decided the basic architecture of our unified system. Now to get a deeper insight into the protocols, we have started our experiments based on reverse-engineering of KaZaA and Skype, whose preliminary results we have provided. More work in the field of reverse-engineering and setting up of the architecture for network simulation will be done in the months to come.

## V. Bibliography


1. Kazaa Homepage - *www.kazaa.com*
2. Jian Liang, Rakesh Kumar, Keith W. Ross - *The KaZaA Overlay: A Measurement Study*, Computer Networks Journal (Elsevier), 2005
3. Nathaniel Leibowitz, Matei Ripeanu, Adam Wierzbicki - *Deconstructing the Kazaa network*, Proceedings of the Third IEEE Workshop on Internet Applications (WIAPP), 2003.
4. Skype Homepage - *www.skype.com*
5. Salman A. Baset and Henning G. Schulzrinne - *Analysis of the Skype Peer-to-Peer Internet Telephony Protocol*, INFOCOM 2006
6. Saikat Guha, Neil Daswani, Ravi Jain - *An Experimental Study of the Skype Peer-to-Peer VoIP System*, IPTPS, 2006
7. Wikipedia Page for FastTrack Protocol - *http://en.wikipedia.org/wiki/FastTrack*
8. Qin Lv, Pei Cao, Edith Cohen, Kai Li, Scott Shenker - *Search and replication in unstructured peer-to-peer networks*, Proceedings of 16th ACM International Conference on Supercomputing(ICS'02), New York, NY, June 2002