INDIAN INSTITUTE OF TECHNOLOGY KANPUR

SURGE 2017

PROJECT REPORT

PEER TO PEER VoIP/MESSAGING

Submitted by
Agrim Bari
Roll No. 150057
Department of Electrical Engineering
Indian Institute of Technology
Kanpur – 208016
Email id: agrimb@iitk.ac.in

Under Guidance of
Dr. Y.N.Singh
Department of Electrical Engineering
Indian Institute of Technology
Kanpur – 208016
Phone No: 0512-2597944
Email id.: ynsingh@iitk.ac.in

CERTIFICATE

This is to certify that the project entitled “**Peer to Peer VoIP/Messaging**” submitted by Agrim Bari (150057) as part of Summer Undergraduate Research and Graduate Excellence 2017 offered by the Indian Institute of Technology, Kanpur, is a bonafide record of the work done by him under my guidance and supervision at the Indian Institute of Technology, Kanpur from 15th May ,2017 to 11th July ,2017

Dr.Y.N.Singh

TABLE OF CONTENTS

ABSTRACT

A peer to peer messaging service code name Brihaspati 4 (B4) Messaging refers to a real-time text transmission wherein exists a set of peer nodes(users) that function both as receiver and as a sender of messages to any other node in the network of nodes, and additionally as recipient of messages of some other node(user). This model of messaging service contrasts with the general architecture wherein there is a fixed hierarchy of routing proxies and user agents. Any node is able to initiate or complete any supported transaction. Each node joining the DHT- based overlay network would be authenticated only once in the start where itself a certificate along with public and private keys will be generated which will act as a proof of being a valid user. Now, when a chat has to be initiated between two users, one user will pass a query in through the communication manager which will be responded with IP Address of the desired user whom to be contacted, and thus public keys and certificates will be exchanged for the mutual authentication process. Once the authentication is done both ends will have a session key to encrypt and decrypt the messages for the online session. In the scenario of the receiver is offline, the successor node will be contacted who will then receive encrypted messages, which could be only decrypted by the desired user with his private key.The messages stored in the successor node will be directed towards the desired user once he comes back online, thereby decrypting them. The messages are transmitted in the encrypted form by making use of socket programming making it a secure channel for chatting against various forms of threats. In all, the distributed system of messaging makes it more reliable and scalable than the centralized versions, secure than the previous existing peer to peer architectures, easily accessible other user data by making use of routing tables and making use of user clouds for storing data against the presence of central clouds.

INTRODUCTION

A peer-to-peer (P2P) network is a distributed application architecture in which interconnected nodes ("peers") self-organize themselves into overlay networks and share resources amongst each other without the use of a centralized administrative system. Peer- to-Peer Systems are built on the foundation of client-server wherein every node in the overlay network serves as a dual functional client and server. P2P systems distribute the main cost of disk space and for storing files and bandwidth for transferring them- across the peers in the network, thus enabling applications to scale without the need for powerful, expensive servers. Their ability to build an extremely resource rich system by aggregating the resources of a large number of independent nodes enables peer to peer systems to dwarf the capabilities of many centralized systems for relatively little cost.

P2P systems can be classified into two different classes: Structured P2P systems and Unstructured P2P systems.

In structured P2P systems, connections among peers in the network are fixed, and peers maintain information about the resources (e.g., shared content) that their neighbor peers possess. Hence, the data queries can be efficiently directed to the neighbor peers who have the desired data, even if the data is extremely rare. Structured P2P systems impose constraints both on node (peer) graph and on data placement to enable efficient discovery of data.The most common indexing that is used to structure P2P systems is the Distributed Hash Tables(DHTs) indexing. Similar to a hash table, a DHT provides a lookup service with (key, value) pairs that are stored in the DHT. Any participating peers can efficiently retrieve the value associated with a given unique key. However, this may result in higher overhead compared to unstructured P2P networks. Different DHT-based systems such as Chord, Pastry, Tapestry, CAN are different in their routing strategies and their organization schemes for the data objects and keys.

There are however important challenges that must be overcome before the full potential of the P2P system can be realized.

Scalability - there is no algorithmic or limitation on the size of the system, thus the autonomy of nodes make it difficult to identify and distribute the resources that are available.

Reliability - some peer might be malicious, peers may receive inauthentic information or may be victims of denial-of-service-attacks.

Peer to Peer VoIP/Messaging

Why Peer to Peer Systems are efficient and preferred more:

When a user wishes to download a file from a website, they submit an HTTP GET request. This request for the file uses a single TCP socket and communicates with a single server which transfers the entire file. By contrast, a P2P protocol creates TCP connections with multiple hosts and makes many small data requests to each. The P2P client then combines the chunks to recreate the file. A single file host will usually have limited upload capacity, but connecting to many servers simultaneously allows for higher file transfers, and disperses the costs associated with data transfers amongst many peers. Moreover, a client mid-way through downloading the file also acts as a server, hosting the bits to others which they have already downloaded. These differences from traditional HTTP GET requests to allow for lower costs and higher redundancy since many people are sharing the files.

These few points clearly state why P2P is preferred more:

1) It is easy to install and so is the configuration of computers on this network,

2) All the resources and contents are shared by all the peers, unlike server-client architecture where Server shares all the contents and resources.

3) P2P is more reliable as a central dependency is eliminated. Failure of one peer doesn't affect the functioning of other peers. In the case of Client –Server network, if the server goes down the whole network gets affected.

4) There is no need for full-time System Administrator. Every user is the administrator of his machine. The user can control their shared resources.

5) The overall cost of building and maintaining this type of network is comparatively very less.

LITERATURE REVIEW

The necessity for P2P systems was based on the idea to get rid of an application’s dependency on the maintenance of a server. With this idea in mind, in 1999 a few applications were developed: the Napster music-sharing system, the Freenet anonymous data store, and the SETI@home volunteer-based scientific computing projects. Napster, for instance, allowed its users to download music directly from each other's computers via the Internet. More than a decade later, P2P gained a lot of research interest and a lot of work has been done in order to use this feature and implements the feature judiciously. Some of the applications built over the years on the same principle are as follows:-

- BitTorrent is one of the most popular peer-to-peer file sharing protocols and it accounts for a significant amount of traffic on the Internet.
- DistriBrute is the world’s first peer-to-peer (P2P) desktop deployment product specially developed for business use. Using DistriBrute it is possible to simultaneously migrate thousands of desktops to a new OS and quickly install new applications and drivers.
- There exist various other file sharing systems that have been developed such as Tribler which acts as an integrated search engine. However, Pando is a personal P2P program, much like BitTorrent but geared toward those looking for a simple and secure means of file transfer. Users may email, IM, or post to their website a .pando file.
- Peercasting, much like broadcasting, it is a method of streaming content to consumers. But it differs from traditional broadcasting because the consumers of peer casted content are simultaneously broadcasters. Freecast is a peer-to-peer streaming audio broadcasting program that has been developed.

Another major use of P2P technology on the Internet is for making audio and video calls, popularized by the Skype application.

Despite having so many applications based on the same system, this project aims to be stricter in terms of security. It is rather similar to the usability of Skype, however, what makes it different from the same as it’s ability to store the offline messages to 5 different peers, the constant updation of the routing table and the usage of both symmetric and asymmetric cryptographic systems. Since there is no server involved, a lot of messages are now no more stored in the server occupying unnecessary space. Having the data being split to different users, a lot of memory is saved in this process.Moreover, the messages are not lost and are easily received by the user as soon as it comes online. Thereby eliminating the dependency of any server and als ensuring that the messages are definitely transferred to the user aimed.

WORK DONE

Dealing with the Queries and Topologies

We have implemented the search techniques based on distributed hash tables (key: UserId , value: hash value) pairs are stored in a DHT, and any participating node can efficiently retrieve the value associated with a given key. Responsibility for maintaining the mapping from keys to values is distributed among the nodes, in such a way that a change in the set of participants causes a minimal amount of disruption. This allows a DHT to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures.

Dealing with the Sending the Receiving of messages

We have made use of socket programming for developing the connection. Four sockets are used one is for the online session during which only when the messages or files are to be sent or received a socket is opened. The second one is used for the case when the other user is offline i.e. to send the messages or files to the successor node during the transmission period only. The third socket is used to receive the messages or files of whom the node is successor node and thereby storing in the user cloud and the fourth one is used for receiving the messages or files from the successor node.

Dealing with the Security issue

We have implemented asymmetric cryptography (RSA algorithm) making use of user's public and private keys for the encryption and decryption of the random strings which are sent to one another, and which is then being concatenated together for the generation of session key, and when the transmission of messages begin, symmetric cryptography(AES algorithm) is used for encryption and decryption where session key is used as the symmetric key. For the offline session, we have made a joint usage of symmetric and asymmetric cryptography whereby only the desired user will be able to decrypt the messages.

Peer to Peer VoIP/Messaging

The stepwise procedure followed for a successful messaging communication between two certified users is explained below:

Step 1 Mutual Authentication and Generation of Session Key

VERIFICATION:

CLIENT A	CLIENT B
Receives Certificate and Public Key from Client B, certifies using the public key of Central Authority	Receives Certificate and Public key from Client A, certifies using the public key of Central Authority
Generate a random string A - encrypt with the public key of Client B.	Generate a random string B - encrypt with the public key of Client A.
Send it to Client B.	Send it to Client A.
Receives random string B and decrypts using the private key of Client A.	Receives random string A and decrypts using the private key of Client B.
Sends random string B back to Client B encrypted with the public key of Client B.	Sends random string A back to Client A encrypted with the public key of Client A.
If string received correctly, Client B is authenticated.	If string received correctly, Client A is authenticated.

GENERATION OF SESSION KEY:

Random string A + Random string B = Session Key

This session key is unique to the sender and the receiver and will be used for the encryption and decryption of messages.

Step 2: Encryption

if (receiver = online)

{Employing Symmetric Cryptography, making use of Session Key for Encryption and Decryption of messages}

else { The sender transmits the messages along with an offline_session key to 1 successor peer of the receiver whose information(IP Address) is obtained from the routing table. The messages are encrypted using the offline_session key, and the offline_session key is encrypted using the public key of the receiver employing asymmetric cryptography, thus the messages which are stored in files can be decrypted only by the receiver. }

Step 3: Sending the messages

if (receiver = online)

{ Making use of socket programming and sending the messages through Input Output Stream}

else

{ sending the files created for each message and offline_session key making usage of socket programming to its 1 successor node, and initializing the flag of the receiver offline to true (that files are there to be retrieved) by the successor node }

Step 4: Receiving the messages

When instant messaging begins files are retrieved from the successor node if the flag is true

- check the successor nodes if there are any messages for the receiver

- retrieves the message files from the successor node

When destination is done retrieving, now open to establishing connection with other users for the online session

Step 5: Decryption

Case 1: When the flag is false and online session is going on decrypt with the session key

Case 2: When the flag is true first decrypt the offline_session key using the private key thereby decrypting the message files

FUTURE WORKS

The system developed so far in the case of offline session transmits the data only to one successor node, in the scenario of both receiver and successor nodes being offline sender will wait for any one of them to come online, making it a waste of computation; thus at the least 5 successor nodes need to be there to whom the files can be sent and message files are then to be sent to all 5 of them either simultaneously or subsequently, and thus a thread needs to be implemented which is to be running continuously in the background for receiving the files. Next, when the receiver is offline and the sender needs to send the message, before sending the message a check is to be made to know if the successors have enough space for storage of the messages or not. If not, the routing table gets updated again along with the index management and the first successive peer does the check again, in the same a spillover data implementation could also be thought about. Thus, for the same, an algorithm needs to be developed and implemented.

CONCLUSION

Peer to peer messaging model developed so is an alternative model to that provided by the traditional Client - Server architecture. The service developed is a decentralized model in which each machine referred to as a peer functions as a receiver, sender of messages along with being a recipient of the message of some other peer. With an aim for dealing with the issues of security and privacy from anonymity, accountability, trust on other peer, reputation, information quality to information preservation.

The architecture has so been developed which requires at each step certificate and presence of public and private keys for user authentication and further functionalities implementation, files are being stored in a successor node in case of the receiver being offline to ensure no data loss, information is transmitted and received in encrypted format which could only be decrypted by the second client on the receiving end through the usage of his private key. Even so if the second client is offline the encrypted messages in the form of files will be received in the successor node which the second client will retrieve from whenever he comes back online and decrypt the same by employing his private key for the generation of offline session key which will then be used for decryption of message files so being received. More so instead of deployment of a central server for storing data, user clouds are used hereby for efficient storage management. The system has so been developed to make network efficient and secure in every sense.

REFERENCES

1. PPay: Micropayments for Peer-to-Peer System, Beverly Yang and Hector Garcia-Molina. *In ACM CCS 2003*
2. Privacy Preserving Indexing of Documents on the Network, Mayank Bawa, Roberto Bayardo Jr., and Rakesh Agrawal. *In VLDB, 2003*
3. SLIC: A Selfish Link-based Incentive Mechanism for Unstructured Peer-to-Peer Networks, Qi Sun and Hector Garcia-Molina. *In ICDCS, 2004*
4. Identity Crisis: Anonymity vs. Reputation in P2P Systems, Sergio Marti and Hector Garcia-Molina. *In IEEE Conference on Peer-to-Peer Computing, 2003*
5. Incentives for Combating Free-riding on P4P Networks (Research Note), Sepandar Kamvar, Mario Schlosser and Hector Garcia-Molina. *In ICPDC, 2003*
6. Addressing the Non-Cooperation Problem in Competitive P2P Systems, Sepandar D. Kamvar, Beverly Yang and Hector Garcia-Molina. *In P2P and Economics, 2003*
7. The EigenTrust Algorithm for Reputation Management in P2P Networks, Sepandar D. Kamvar, Mario T. Schlosser and Hector Garcia-Molina. *In WWW, 2003.*
8. Query-Flood DoS Attacks in Gnutella, Neil Daswani and Hector Garcia-Molina. *In ACM CCS, 2002.*
9. Peer Pressure: Distributed Recovery from Attacks in Peer-to-Peer Systems, Pedram Keyani, Brian Larson and Muthukumar Senthil. *In IFIP Peer-to-Peer Computing, 2002.*
10. Protecting the PIPE from malicious peers Brian Cooper, Mayank Bawa, Neil Daswani, and Hector Garcia-Molina. *In FGCS 2004.*
11. Evaluating GUESS and Non-Forwarding Peer-to-Peer Search, Beverly Yang, Patrick Vinograd and Hector Garcia-Molina. *In ICDCS, 2004*
12. Maximizing Remote Work in Flooding-based Peer-to-Peer Systems, Qi Sun, Neil Daswani and Hector Garcia-Molina. *In DISC, 2003*
13. Ad hoc, Self-Supervising Peer-to-Peer Search Networks, Brian Cooper and Hector Garcia-Molina. *Submitted for Publication.*
14. SIL: Modeling and Measuring Scalable Peer-to-Peer Search Networks, Brian Cooper and Hector Garcia-Molina. *Submitted for Publication.*
15. Studying Search Networks with SIL, Brian Cooper and Hector Garcia-Molina. *In IPTPS, 2003*
16. Partial Lookup Services, Qixiang Sun and Hector Garcia-Molina. *In ICDCS, 2003.*
17. Designing a Super-Peer Network, Beverly Yang and Hector Garcia-Molina. *In ICDE 2003.*

Peer to Peer VoIP/Messaging

18. Efficient Search in Peer-to-peer Networks. Beverly Yang and Hector Garcia-Molina. *In ICDCS, 2002.*
19. Routing Indices for Peer-to-peer Systems. Arturo Crespo and Hector Garcia-Molina. *In ICDCS, 2002.*
20. Comparing Hybrid Peer-to-Peer Systems. Beverly Yang and Hector Garcia-Molina. *In VLDB, 2001*