STANDARDS-BASED P2P COMMUNICATIONS SYSTEMS

David A. Bryan
Bruce B. Lowekamp, advisor
Department of Computer Science
College of William and Mary

Abstract

Current Voice over IP (VoIP) and Instant Messaging (IM) systems follow a client-server model, or require clients and network elements supporting non-standard protocols. In this paper, we present a standards-based, fully decentralized peer-to-peer (P2P) communications system. This system can be used in situations where constant Internet connectivity is not possible, and supports ad hoc, ephemeral communications scenarios. These systems are useful in environments including remote locations, security conscious organizations, and collaborative groups. Our system leverages the existing SIP/SIMPLE family of open-standard protocols, ensuring that our system will interoperate with existing communications infrastructure. We implement our system using a Distributed Hash Table (DHT) approach for lookup, transported using SIP messages. We support replication for reliability and availability. Our system uses unmodified SIP/SIMPLE mechanisms to exchange messages after resources are located, allowing for maximal reuse. We explore future directions of research for such systems, including security, availability, and advanced routing, including social-based routing techniques.

1 Introduction

There are many mechanisms that are used for communications between individuals, and today, Voice over IP (VoIP), as well as Instant Messaging (IM) are very popular. These technologies can be extended to handle video/vision, resource sharing, and even remote control. In addition to the obvious uses for communications, the protocols for these technologies are being extended for many other applications. There is growing interest in using this type of technology to aid in remote rehabilitation of injured persons out of reach of medical care [23]. Additionally, this technology has been explored for use in distributing information and communication for air traffic control [4], wearable/on person computing environments [1], document sharing, communications within simulators, and many other purposes.

While these systems are popular, and are becoming increasingly so, a number of issues limit deployment in some scenarios. Most of these systems use a client-server architecture. That is, these systems consist of a number of client programs, run by the user, that connect to one or more centralized servers. This can be a problem for users, particularly for security reasons or when access to the servers is limited. Deploying private systems can address security concerns, but interacting with outside users can then become an issue. Additionally, the resources needed to deploy a private system often outweigh the advantages. Deploying servers for ad hoc groups or in remote locations can be impractical if not impossible.

We have developed a system we call SOSIMPLE to address these issues. We have taken a well document, industry-standard communications protocol for VoIP and IM (the SIP/SIMPLE [15] family of standards) and extended it using Peer-to-Peer (P2P) technology, so that user's access devices directly communicate with each other. The particular P2P technology used is a Distributed Hash Table (DHT). This design address many problems, while offering compatibility with existing Commercial Off The Shelf (COTS) equipment in use for communications today, including IP telephones, IM clients, and gateways (devices to connect IP systems to public phone lines). Additionally, since we leverage an existing protocol, our design is well supported by existing network infrastructure such as firewalls, which can recognize our traffic as communications traffic and treat it appropriately. As a step in encouraging adoption, and ensuring compatibility, we have submitted a draft for a P2P SIP protocol based on SOSIMPLE [2] to the Internet Engineering Task Force (IETF), the body responsible for standardizing protocols on the Internet.

This paper motivates our work by discussing scenarios in which traditional client-server approaches to communications are not appropriate. We then discuss the current state of VoIP and IM systems, and discuss related work. This information is used to derive requirements for a replacement system. Finally, we offer our P2P SIP based approach as a protocol meeting the requirements for a decentralized, next-generation communications infrastructure.

2 Motivating Scenarios

There are a number of scenarios where traditional client server communications systems fail or perform poorly. These include:

- **Limited or No Internet Connectivity:** There are many reasons a user may not have external connectivity. If the user is located in a very remote location (perhaps Antarctica, a remote developing country, or even another planet), Internet connectivity could be intermittent, delayed, or non-existent. Similarly, there may be reasons that connectivity is interrupted, ranging from a failure at the service provider to being onboard an aircraft. Finally, security, such as being located in a classified environment can preclude outside connectivity.

- **Ad-hoc and Ephemeral Groups:** Groups that collaborate may be ephemeral, such as in a meeting, classroom, field research, battlefield, or conference setting. These users should be able to communicate and collaborate without having to configure a server.

- **Small Organizations:** Small organizations may want private, internal communications, but may not be able to afford the equipment or expertise needed to install and maintain it. Public services such as AOL's AIM, Microsoft's Messenger, and the Vonage VoIP service, use servers located at the provider, compromising security.

- **Censorship or Impeded Access:** Some users may not be allowed to access centralized, persistent servers because of government censorship or an access provider that prevents access to competing services.

- **Scalability:** Centralized systems do not scale well as users are added. Such systems continually need to add and integrate additional servers in order to add users. A decentralized system utilizing standards-based equipment should be more reliable, easier to join, and ultimately less expensive.

3 Background

3.1 VoIP and IM

Most VoIP systems are based on either the older H.323 protocol, or the newer SIP [15] protocol from the IETF. Systems deployed today are largely client-server. Users connect using a User Agent (UA). UAs can take the form of a software application or a hardware device such as an IP or mobile phone. The UAs connect to a central sever, usually called a proxy, softswitch, or gatekeeper. H.323 uses central control, while SIP places much functionality in the UAs. A large outlay of effort and capital has been invested in building networks based on these protocols. Most commercial “hard” devices for VoIP, such as IP telephones and gateways are designed for H.323 or SIP.

The IM protocols used today, such as AOL’s AIM, Microsoft’s MSN Messenger, and Yahoo’s Y! Messenger use a central server located at the providers location. Some commercial products as well as the emerging IETF standards XMPP and SIMPLE [3, 13, 18] are designed to be deployed by enterprises. Many organizations have banned the use of commercial IM products because private conversations flow through a third party location. The overhead of an organization maintaining their own IM servers in-house often exceeds the benefits gained from IM.

VoIP and IM systems provide for resource location, session establishment and management, and presence. Resource location is responsible for identifying and locating other users so conversations can be established. Session establishment and management establishes, manages and terminates text or multimedia sessions between users. Finally, presence refers to the ability of users to determine if other users (such as a group of “buddies” or “friends”) are connected, and to be notified when they arrive or leave. A user may have a list of individuals with whom they wish to communicate. These users are called “friends” or “buddies.” Presence allows the user to see when a friend has entered, perhaps by highlighting that user on a list displayed on the user’s GUI.

3.2 SIP and SIMPLE

SIP and SIMPLE are text-based protocols derived from HTTP [6]. As mentioned earlier, SIP is a general protocol for establishing and controlling multimedia sessions. SIP can establish voice sessions, video sessions, remote collaborative/control sessions—essentially any situation where one needs to establish and maintain a session of information between two or more users. SIMPLE is a set of SIP extensions allowing for IM. IM “sessions” generally are more simple than a full SIP session. One simply sends a message to another user. It is up to the user’s client to correlate those messages into a conversation. In every other way, SIP and SIMPLE behave similarly, but we will note other differences where appropriate. When used for voice or video, SIP embeds Session Description Protocol (SDP) [8] describing the media.

A central server is generally used to help with resource location. This device performs several functions described in the SIP standard, including registration (storing information about where users are located), and routing the calls between users. This entity is often referred to simply as a proxy, even though it has more functionality and complexity than a web proxy. Signaling messages to establish the session are sent from the calling UA to the proxy, which locates the other UA and passes messages to it, possibly through additional intermediary proxies. Proxies are also responsible for authenticating UAs and administering user names. Proxies are located either on site or at a central service provider. For efficiency, the media is streamed directly between the UAs, rather than through a proxy, usually using Real Time Protocol (RTP) [19] packets.

Every user has a SIP URI that identifies them, usually of the form “sip:bryan@cs.wm.edu”. Users send a SIP REGISTER message to the proxy (acting as a registrar), which stores the mapping between the URI and the user’s UA’s IP address. The REGISTER message is resent periodically to refresh these entries, which expire after a time. Users can also remove their registrations.

SIP uses the INVITE message to establish a new session. This message is sent to the proxy, which looks up the destination user and forwards the message to the destination UA. If the proxy cannot locate the person the session is to be established with, the call may be sent to another proxy, to the public phone system, or rejected. Sessions are terminated with the BYE message. Instant messages are carried in a message with the confusing name of “MESSAGE”. This message contains the text of a text message—it does not establish a new session.

When trying to establish a session, response codes are returned to the caller to let them know the result of the attempt. These results can provide answers about the result of the call setup (such as 200 OK, 404 Not Found or 302 Moved Temporarily), or simply provide status updates, such as 100 Trying or 180 Ringing.

SIMPLE and SIP also offer SUBSCRIBE/NOTIFY functionality. This allows a device to subscribe to a string. When the string "occurs" (as defined by the device that has been subscribed to), a NOTIFY message is sent to the subscriber. This is used in SIP for services such as voice mail indicator lights. In SIMPLE, this mechanism is used to implement presence. This mechanism can also be used when controlling devices to pass real time information about events between the device and the user.

Because SIP is an open standard, many freely available open source implementations of stacks and applications are available, and much existing equipment is installed. SIP has emerged as the standard protocol for VoIP and IM, and so most new development in the space is being done in SIP. Because of this, we choose SIP rather than another protocol such as H.323 or inventing our own standard.

3.3 P2P Systems

Peer-to-Peer (P2P) systems are very different than client-server systems. Each node participating in such a system is of equivalent importance. In addition to participation as a client, each node provides some server-like functionality as well, providing services to other nodes periodically, and receiving services from those nodes in exchange. The web of peers formed in such a network is referred to as an overlay network, as the connections between peers overlays the underlying physical network. P2P networks are transient in nature, with nodes entering and leaving freely and frequently. To address this transient nature, these networks usually incorporate adaptability and replication.

P2P systems can be used for many purposes, the most famous (or infamous) being file sharing. They are broken into two broad categories, unstructured and structured. Unstructured systems typically utilize an approach referred to as flood search, in which all nodes are queried to locate the one providing a resource or service. Structured P2P systems organize the nodes so that specific resources or services can be located in a more direct fashion, querying only a few nodes. One particular method called a Distributed Hash Table (DHT) uses hashed keys to store resources. A key associated with the resource is hashed, and each node also has an ID in the same hash space. The node with the key nearest the resource's hashed key is responsible for that resource. A particular node knows about a few other nodes, and asks the node they know nearest to the resource to locate it. That node in turn asks the nearest node it knows. The process repeats until the resource is located. Chord [22] is a DHT implementation using a ring structure to organize the information.

Many systems, such as the original Napster system, many existing SIP deployments, and Skype [21], discussed below, are what are referred to as hybrid systems. In these systems the resource location information is stored by a central server, and the resources (the media, in the case of SIP) are distributed. This differs from a true or pure P2P approach, where the location of the resources is also distributed.

Locating the first node in the overlay to get started is a difficult problem faced by all P2P systems. It is generally addressed either using an out-of-band mechanism to locate a "bootstrap" node, or by using a multicast/broadcast mechanism.

P2P systems generally scale better (since each node that joins also brings additional services), provide more resistance to failure of a single node, and provide a measure of protection against denial of service (DOS) attacks. Such systems face an assortment of problems, however, including attacks in which some corrupt node fails to forward requests or provides incorrect results, and attacks of saturating the network with many duplicate identities. Such an attack is referred to as a Sybil attack. [5]

4 Related Work

Skype is a VoIP/IM system allowing users to make free phone calls between Internet users, and offering public phone connectivity for a fee. The system is proprietary, and is not compatible with non-Skype devices. Skype is a hybrid system, where username control, authentication, and some other services are controlled by Skype's central servers.

The EarthLink R&D group has created an application called SIPshare [20], using SIP as the underlying transport protocol for a file-sharing P2P system. Similarly, NUTSS [7] is a project exploring how using SIP and related technologies might be useful as a general mechanism for transport in P2P networks, and in particular for NAT traversal. In contrast, SOSIMPLE uses P2P to decentralized a SIP/SIMPLE communications system.

Singh and Schulzrinne are exploring a concept similar to ours [11]. Key differences in our approach include different SIP mechanisms underlying the system, a recursive, rather than iterative approach to P2P search, and using friend list information for routing.

5 Requirements

The following features are required to solve the problems we have outlined.

- **No Central Server/Connectivity:** In several scenarios, security or connectivity prohibit contacting an outside server, therefore we require none be present.

- **User Mobility Support:** Users should be able to move from device from device (from a mobile device in the field to a device in an office, for example) and maintain access to configuration information, friend lists, etc.

- **No Central Naming Authority:** No central authority should involved in naming. Users should be able to choose names and the P2P network should mediate disputes over these names.

- **Simple System Discovery:** For scenarios where appropriate, a simple mechanism must be available for locating the system, for example a broadcast mechanism.

- **Scalable Number of Users:** Additional users must be able to join the system, and the system should grow in response. No new resources, other those the new user brings, should be required.

- **Privacy:** Users should be able to locate other users without having to go outside the system. Messages between users must not leave the system or be interceptable by other users within the system.

- **Multiple Realms:** Users should be able to configure their own instance of a communication system and limit access. We refer to each such instance as a *realm*. A security conscious group would typically utilize a private realm for internal messages, and interact with external realms for outside communications..

- **Interconnection:** While traffic within a realm should be restricted to that realm, we must support interconnecting independent, separate realms.

- **Abuse Prevention:** The system should make an effort to prevent abuses such as harassing communication, or misuse for activities such as file sharing.

- **Compatibility and Reuse:** The system must be compatible with as much existing infrastructure as possible, including VoIP and IM clients, and VoIP gateways. Code should be reused where possible.

6 P2P SIP-Based Solution

Our solution takes a P2P approach to SIP, resulting in a distributed communications system with no central server. We selected SIP because it is an established standard, in which most new VoIP and IM development is being done. It also allowed us to leverage and reuse as much existing technology as possible.

Our desire to remove dependence on servers or Internet connectivity, as well as high scalability, led us to take a a P2P approach. While dynamic DNS approaches can meet some of our goals, they often introduce lags in user mobility. Additionally, in order to handle scenarios where outside access is not possible, we ruled out a DNS based approach. P2P seemed the natural choice. We decided to use SIP as the underlying transport for our P2P traffic as well, since there is no clearly established transport for P2P systems. Finally, since many border devices (firewalls, traffic shapers, etc.) are already able to recognize SIP traffic—as well as recognizing and often banning P2P traffic—using SIP as the underlying mechanism will ease deployment.

7 The SOSIMPLE P2P SIP Architecture

Because our design has no central servers and nodes communicate directly with one another, there are some differences between our approach and traditional SIP. Peers connect to a few other peers in the overlay network and use these nodes as the destination for most SIP messages. Nodes act both as UAs and as proxies simultaneously. Collectively, the peers replace the functionality of SIP registrars and proxies—each node being responsible for those roles for some portion of the overlay.

7.1 P2P Overlay Structure

Nodes are organized in a DHT based on Chord. Each node is assigned a Node-ID which is created by hashing (using SHA-1) the real IP address (found using a mechanism such as STUN [16] if the node is behind a NAT) and appending a port. We use the same algorithms used by Chord to maintain our overlay network, but as shown below, we use SIP messages to implement the DHT operations. We implement redundancy in the same way that Chord implements it, storing resources on the node responsible for a particular resource, as well as $k = 4$ successor nodes.

SOSIMPLE nodes maintain a smaller number of finger table entries than in Chord. Chord attempts to provide rapid resource location and assumes that users will query many times for many distinct items. To support this, Chord keeps 160 entries in their finger tables. While we desire reasonable resource location time, users often message or call the same user many times (for example co-workers, friends or family members). Additionally, delays of even a full second are insignificant when setting up a phone call or IM session. As such, we maintain only 16 finger table entries for routing, but prefetch entries for contacts in the user's friend list, as well as caching previously contacted nodes.

7.2 Message Format

All messages for both DHT and user operations are SIP messages. SIP was designed to be extensible, and so we have been able to implement these messages using only new headers for the SIP REGISTER method—no new methods are required. We use the REGISTER message to pass the DHT information between nodes. Our architecture makes a distinction between two ways in which we use REGISTER messages. Use for the original purpose of sending user location information to a registrar is referred to as *User Registration,* while use for DHT operations, including entering and leaving the overlay, and exchanging node location information is referred to as *Node Registration*.

7.3 DHT Operations

A node wishing to join the overlay locates an initial node, referred to as a bootstrap node. Presently this node is located via out-of-band mechanisms. The joining node sends a REGISTER with the joining node's Node-ID (computed as above) to the bootstrap. The bootstrap responds with information about nodes that will be near the joining node when it is accepted to the overlay. Additional messages convey information about neighbors, finger table information, etc. This information is conveyed in headers within REGISTER messages. Since the joining node may now be responsible for some user registrations that were formerly stored by other nodes, a number of user registrations may also be forwarded to the new node from existing node(s). Please see our IETF Internet-draft [2] for details of the exact format of the SIP messages.

7.4 User Operations

Registration and resource location are decentralized to meet our requirements for *No Central Server* and *No Central Naming Authority.* The registration process is modified by changing where registration messages are sent.

After a node has joined the overlay, the user or users associated with that node must be registered (user registration), creating a mapping between a SIP URI and the IP address of the user's UA. The user name is hashed to produce a Resource-ID. The REGISTER message for the user is sent to the finger table entry nearest this value in the overlay, rather than a fixed server as in a traditional client-server model. If the node receiving the message is not responsible for that value, it will refer the sending node to a closer node from the receiver's finger table. This process is repeated until the node responsible for storing the registration is found, at which point that node stores the information.

Session establishment works in much the same way. The calling node will hash the name of the called and send the INVITE or MESSAGE to the node in the finger table nearest the hashed name. Redirects will be used to close in on the responsible node, which will send a final redirect, this one referring the caller to the actual UA of the called. The contact will be cached for future use.

Once the UA is located, IM traffic can be sent between the nodes at the SIP/SIMPLE level using the MESSAGE method. The overlay need only be involved when a message is first sent or if a cached contact fails, subsequent messages will be sent directly to the cached address. The signaling messages needed to establish and terminate a voice session are transmitted over the overlay, but as in conventional SIP, the media flows directly between the UAs.

In Figure 1, we show an example of the messages passed to perform the distributed search establishing a call between Alice and Bob. Alice's node (which is also her UA), hashes Bob's ID and determines that Node A is the node in her finger table nearest to the hashed value. Alice sends an INVITE to Node A. Node A doesn't store the registration for Bob, but knows that Node B is closer to the hashed value for Bob, so it returns Node B's address to Alice. Alice then tries Node B, which suggests Node C as a closer node. Node C does store the registration information for Bob's UA/Node, and returns it to Alice. Alice uses this information to contact Bob.

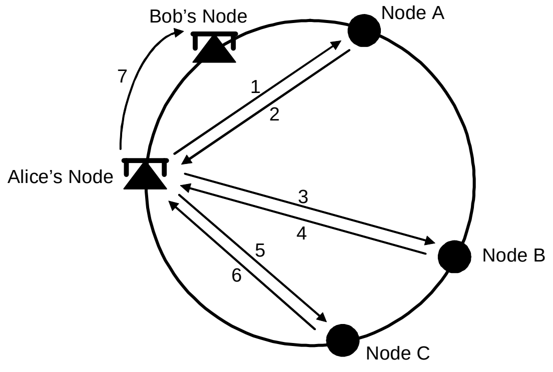

Figure 1: Locating a user

When a P2P UA starts, the SIP SUBSCRIBE/NOTIFY mechanism is used to attempt to establish subscriptions with each person in the user's buddy list to track their availability and prefetch their addresses. The friend list is stored as an encrypted file within the overlay, to meet our requirement for *User Mobility Support* .

8 Current Research and Future Work

There are several key areas of research that arise from designing a P2P SIP-based communications system. We have explored a number of these, and have found a rich vein of future research.

8.1 Security and User Authentication

Security is a key research question. SIP/SIMPLE uses private-key mechanisms for authentication between UAs and proxies and for end-to-end encryption [10]. A P2P architecture requires a public key system that assumes no trust between nodes. We are currently evaluating a number of possible approaches. Ultimately, we expect that more than one approach will prove desirable, and different instances of P2P communications systems will weigh the relative strengths and weaknesses of different approaches and choose the one most appropriate for the deployment. In the case of ad-hoc or transient networks, no verification may be required or desirable. The mechanisms we have explored include:

- Storing the public portion of a user's key in the overlay. This serves to ensure that user is who they say they are the second time communication is established, but makes no guarantees for the initial connection. Such a mechanism provides no name space conflict guarantees, and is best combined with other mechanisms.

- Email verification of user identities requires user names be valid email addresses. Registering users are challenged by the node handling the registration using email. User names are guaranteed to be unique, but a malicious node can deny access to the system to users for which it is responsible (although since it cannot control where in the hash space it is assigned, it cannot target a particular user). Additionally, outside email access is required for the initial connection—a serious drawback.

- The PGP web of trust model is also an alternative. In such an approach, a user would have a certificate signed by one or more other users of the system. If a path of signatures leads to a trusted node, the new node is considered trusted. This can be run globally or within the nodes of the overlay only.

- Centrally managed certificate stores using a mechanism such as X.509 can be used, resulting in a system much like the system currently used for email signatures and encryption. The root certificates can be compiled into the client, the result being that one needs Internet access only to obtain a certificate, not to communicate.

- Attacks involving multiple identities such as the Sybil attack [5] can be mitigated using computational puzzles to limit the rate at which new identities are issued. Registrations must be accompanied by a solved puzzle, which is associated with the user ID that is being created. Many alternatives for such puzzles exist, one need only ensure that they can be verified trivially. Jennings has proposed such a standard for SIP clients in a recent Internet-draft [9].

None of these mechanisms are currently standard SIP, and must be implemented between our modified nodes. There are efforts underway within the IETF to standardize end-to-end authentication and we will attempt to incorporate these as they emerge.

8.2 NAT Traversal

NAT traversal has been well explored in the IETF in the STUN and TURN drafts. [14, 16] The problem of how to implement this in a purely P2P solution lies in determining if it is possible to distribute the function among the nodes, or of public STUN/TURN servers will be required.

8.3 Routing

There are a number of issues relating to routing we are currently exploring.

- We are using simulation to determine parameters for our finger tables. Early results indicate P2P communications systems need fewer fingers than file sharing, particularly if friend/contact list entries and recent contacts are cached.

- Social routing is another concept being explored. SPROUT [12] explored using social routing for

P2P. We are investigating how this applies to communications systems. Because users frequently have buddies on their lists that have similar interests, a communications system should be an ideal candidate for a P2P approach combining DHT and social based routing. We are simulating the effectiveness of these techniques.

- The underlying P2P architecture is also an area that we are currently evaluating. While we presently use a system very much like Chord and are evaluating social network extensions to this, we also are working to evaluate other P2P mechanisms, such as those used in Pastry or Tapestry [17, 24].
- Interconnection of multiple realms and with non-P2P aware SIP legacy systems in an efficient way is a topic we have just begun to explore fully. .

8.4 Persistent Storage and Abuse Prevention

One final area of future research is investigating persistent storage. Persistent storage is needed for a number of purposes—storing public keys, storing friend lists, voicemail, and storing configuration information. These functions must be supported without allowing the use of SOSIMPLE as a mechanism for illicit file sharing.

9 Conclusion

We have worked carefully to ensure that SOSIMPLE meets the requirements for a practical P2P VoIP/IM system. We have removed dependence on a central proxy server while offering compatibility with as much existing equipment as possible. We have taken an approach to security that allows for authentication without requiring a fully secure P2P system. In short, SOSIMPLE meets the need for a more distributed communication system.

References

[1] A. Acharya, S. Berger, and C. Narayanaswami. Unleashing the power of wearable devices in a SIP infrastructure. In *3rd IEEE International Conference on Pervasive Computing and Communications (PerCom) 2005*, Kauai, Hawaii, USA, Mar 2005.

[2] D. A. Bryan and C. Jennings. draft-bryan-sipping-p2p-00. `http://www.ietf.org/internet-drafts/draft-bryan-sipping-p2p-00.txt`, Jan. 2005.

[3] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle. RFC 3428 - session initiation protocol (SIP) extension for instant messaging. `http://www.ietf.org/rfc/rfc3428.txt`, Dec. 2002.

[4] K. Darilion, C. Kurth, W. Kampichler, and K. M. Goeschka. A service environment for air traffic control based on SIP. In *37th Hawaii International Conference on Systems Sciences*, Big Island, Hawaii, USA, Jan 2004.

[5] J. R. Douceur. The sybil attack, Mar 2002.

[6] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC 2616 - hypertext transfer protocol - HTTP/1.1. `http://www.ietf.org/rfc/rfc2616.txt`, June 1999.

[7] S. Guha, Y. Takeda, and P. Francis. NUTSS: A SIP-based approach to UDP and TCP network connectivity, Aug 2004.

[8] H. Handley and V. Jacobson. RFC 2327 - SDP: session description protocol. `http://www.ietf.org/rfc/rfc2327.txt`, Apr. 1998.

[9] C. Jennings. draft-jennings-sip-hashcash-01. `http://www.ietf.org/internet-drafts/draft-jennings-sip-hashcash-01.txt`, Feb. 2005.

[10] C. Jennings, J. Peterson, and M. Watson. RFC 3325 - private extensions to the session intiiaion protocol (SIP) for asserted identity within

trusted networks. `http://www.ietf.org/rfc/rfc3325.txt`, Nov. 2002.

[11] K. Singh and H. Shulzrinne. `http://www1.cs.columbia.edu/~kns10/research/p2p-sip`.

[12] S. Marti, P. Ganesan, and H. Garcia-Molina. Sprout: P2p routing with social networks. In *First International Workshop on Peer-to-Peer Computing and Databases (P2P&DB 2004)*, Mar 2004.

[13] J. Rosenberg. RFC 3856 - a presence event package for the session initiation protocol (SIP). `http://www.ietf.org/rfc/rfc3856.txt`, Aug. 2004.

[14] J. Rosenberg, R. Mahy, and C. Huitema. draft-rosenberg-midcom-turn-07. `http://www.ietf.org/internet-drafts/draft-rosenberg-midcom-turn-07.txt`, Feb. 2005.

[15] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. RFC 3261 - SIP : Session initiation protocol. `http://www.ietf.org/rfc/rfc3261.txt`, June 2002.

[16] J. Rosenberg, J. Winberger, C. Huitema, and R. Mahy. RFC 3489 - STUN: simple traversal of user datagram protocol (UDP). `http://www.ietf.org/rfc/rfc3489.txt`, Mar. 2003.

[17] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. *Lecture Notes in Computer Science*, 2218:329–??, 2001.

[18] P. Saint-Andre. RFC 3920-3923 – IETF XMPP RFCs . `http://www.ietf.org`, Oct. 2004.

[19] H. Shulzrinne, S. Casner, R. Frederick, and V. Jacobson. RFC 1889 - RTP: a transport protocol for real-time applications. `http://www.ietf.org/rfc/rfc1889.txt`, Jan. 1996.

[20] SIPShare Project. `http://www.research.earthlink.net/p2p/`.

[21] Skype Technologies, S.A. `http://www.skype.org/`.

[22] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of ACM SIGCOMM 2001*, pages 149–160. ACM Press, 2001.

[23] J. M. Winters. Telerehabilitation research: Emerging opportunities. *Annual Review of Biomedical Engineering*, 4:287–320, 2002.

[24] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. Tapestry: A global-scale overlay for rapid service deployment. *IEEE Journal on Selected Areas in Communications*, 2003.