Efficiency of Social Connection-based Routing in P2P VoIP Networks

Vikram Chandrasekaran[1], Ram Dantu[1], Neeraj K Gupta[2]
Department of Computer Science
University of North Texas
Denton, TX- 76203
{vc0063, rdantu}@unt.edu[1],ngupta94@yahoo.com[2]

Xiaohui Yang[3], Duminda Wijesekera[4]
Department of Computer Science
George Mason University
Fairfax, VA- 22030
xhyanggmu@gmail.com[3], dwijesek@gmu.edu[4]

Abstract— **We have proposed using Social Hashing as a means to provide trust relations and routing in Peer-to-Peer VoIP networks. Social hashing helps in using the social trust relationships to thwart spam and DoS attacks against peers. Social hashing also helps in routing in P2P systems by facilitating a node to find its peer node(s) that contains a required data item, referred to as the lookup problem. Distributed Hash Tables (DHTs) solve this problem by storing (key, value) pairs for data so that data items can be found by searing for a unique key for that data. These tables are stored in several nodes in a P2P network and the (key, value) pairs are updated periodically. In this paper we evaluate the efficiency of Distributed Hash Table (DHT) based social hashing over non social hashing in a series of experiments.**

Keywords- P2P VoIP Systems, P2P routing, Social-network based routing, Social networks

I. INTRODUCTION

Peer to Peer communication has its own advantages over the centralized systems like Improved Scalability, Interoperability, Anonymity, Self organizing capability, etc. In a server-less VoIP system, every node has to find its peers that are able provide it with a required service, usually characterized by the kind of data that is required to complete a task at hand. Examples could be storing voice mail, providing emergence services or completing a connection to a known callee. These tasks are usually characterized by some service attributes (such as to connect a call) and some application specific parameters (such an identifier of the callee). Given the nature of the required service, every node needs to look up peers that may provide such service. The distributed hash table (DHT) provides such a lookup by storing (data value, key) pairs, where the data value part should match the required service's attributes and the key part is an attribute that identifies another node that is willing to provide the required service.

In the past we have advocated using the social network of caller and their callees as a means to build a social-network based P2P VoIP system [1,2,3]. The rationale behind such a design would be based on the premise that most subscribers call and receive calls from other people who have some societal reason (such as family or friends, service provider, work related relationships, social relationships) to call each other. Given this premise, it ought to be possible to use the same relationships to route the societal relationships established or natured using the telephone to route the telephone calls. For example, if Alice came to know of a good plumber Bob because her co-worker Cindy recommended Bob, it should be possible for Alice to use Cindy's contact list to originate a call to Bob. The objective of this paper is to determine the efficiency of such a look up against a general yellow page like look up. In order to do so, we propose extending distributed hash table to be indexed by the social relationships, that we refer to as Social Distributed Hash Tables (SDHTs) and verify their efficiency in an experimental testbed.

In addition to routing, SDHTs can also be used as a means to estimate trust in a P2P VoIP system. For example, when Alice's (socially) smart phone is informed of an impending call, the phone may want to verify the caller's identity in order to determine that the call is not SPAM, and that Alice is not a victim of a DoS attack. That can be partly verified by computing the social relationships that the declared caller may have with Alice by computing the reflexive transitive closure of the dyadic social relationships that may exist between Alice and the caller. But this paper does not focus on this aspect.

The rest of the paper is written as follows. Section 2 describes social routing, and supporting evidence for social routing obtained by analyzing call logs. Section 3 summarized the proposed architecture for our proposed P2P VoIP network. Section 4 describes our experimental testbed and Section 5 analyzes the experimental results. Section 5 has our concluding comments.

II. SOCIAL ROUTING

Fig. 1 shows some potential social connections that can be established and maintained by using the telephone. Figure 1 shows that some subscribers communicate among each other more so that with subscribers outside of the group, making socially close groups. Some subscribers in such socially close clusters may not be communicating infrequently with other clusters, creating collections of socially distant clusters. Conversely those clusters of subscribers that do not communicate with others very much form isolated clusters.

Thus the communicating patters between telephone customers can be modeled by using a so call contact graph

between them, producing an online social network among them. The degree of a node in this contact graph is the number of social contacts of each node. Like online social networks, links in this contact graph are directed and their links show a significant degree of symmetry as shown in fig. 1.

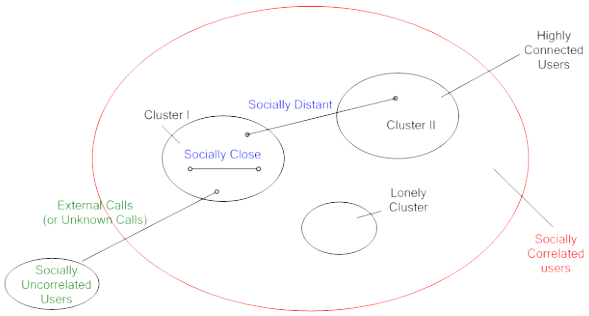

Figure 1. Social Clusters among Telephone Customers

We propose to use society based Node ID assignment and introduce the social hash tables for the storage and routing. We do so by intelligently assigning a numeric identifier [1, 2^{160}] to each telephone customer. Treating each identifier as a sequence of digits of base 2^b, and the ID space is divided based on the customer's social cluster. Members of the same cluster share the same prefix digits, and the hub node is assigned the lowest identifier in each cluster. In addition to leaf set and routing table, we introduce a table for each node, which lists of user's contacts and their corresponding degrees in the social connection graph. We call this table a social table that we use for routing. This process is shown in fig 2.

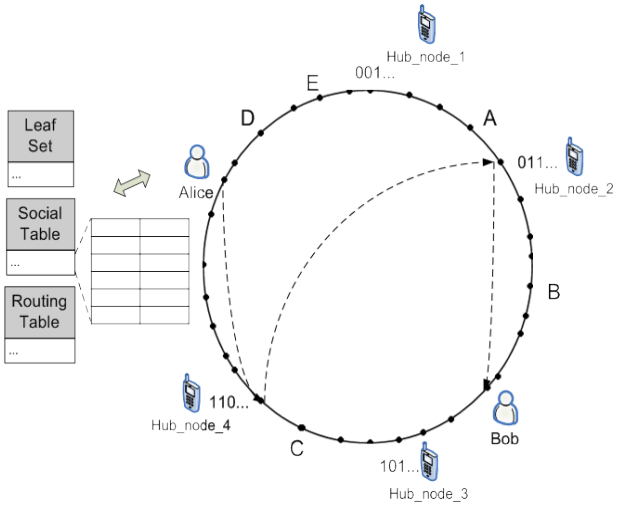

Figure 2: Social Clusters and Social Tables

To route a message with key k, the node needs to check k's existence in a sequence of leaf set, social table, routing table. Social table is used to achieve intra-cluster and inter-cluster routing where nodes that serve as the common connection between two clusters provide the mediated short path between other edges. Based on it, social distance SD_{ij} from node i to j can be calculated as follows:

$$SD_{ij} = \begin{cases} \mu * \sum \left(1 - {d_{k_m}}/{d_{h_1}}\right), i, j, k_m \in C_1 \\ (SD_{ih_1})_{Min} + SD_{h_1h_2} + \left(SD_{h_2j}\right)_{Min}\ i \in C_1, j \in C_2 \end{cases} \quad (1)$$

Where μ is a constant unit distance factor, is the degree of each intermediate node k 1..n on routing path, and h1, h2 is the hub node of cluster C1, C2 respectively. Meanwhile, message should be routed based on (3):

$$D_{ij} = Min\{SD_{ij}, R_{ij}, (SD_{ik} + R_{kj})_{Min}\} \quad (2)$$

where R_{ij} is the routing table based routing distance, and the choose of k is the guarantee for O(log n) efficient routing.

III. Network Architecture

The network stack of our architecture is shown in Figure 3. As seen from Figure 3, our network stack consists of different layers of abstraction for communication. The system consists of a P2P DHT network that contains and maintains the peers for connectivity and the routines for making the routing decision based on the peers in the DHT. The Social Network layer integrates the social telephone information with the DHT network to create the peers for the social network. The Social Hashing is mainly used for routing and estimating trust. Hence we use a social cluster based Node ID assignment and contact oriented social table for lookups.

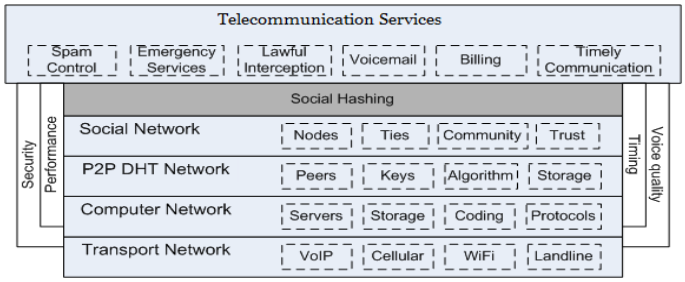

Figure 3. Network Architecture

IV. Our Experimental Testbed

The server-less nature of P2P architecture requires that most intelligence be located in end devices, requiring intelligent P2P phones. We use Ekiga, an open source VoIP application, as our P2P phones. Ekiga supports many audio, video codecs, SIP, RTP protocols, DHT routing algorithms and many kinds of overlay structures.

Based on call setup work flow, we present an architecture design of P2P VoIP phone in Figure 4, which consists of four interacting components: DHT, SIP, RTP, and a Controller. As the core component of test bed, the Controller provides a set of external interface for user interactions with other three components. The controller initializes the P2P phones so that all components have relevant data when they start running, and

ensures that there is a well-configured DHT node available for the use in the P2P overlay network. After initialization, the user can handle the phone call through the Controller, depending on the working scenario of the P2P phones. For an incoming call, the controller notifies the user to take an appropriate action like setting up, hanging up, or transferring the call to another entity, or direct it to an appropriate storage facility. For an outgoing call, the controller queries any DHT node to inquire the presence of the destination entity. If present, the DHT nodes will reply with the intended destination's SIP address. If the destination entity is not present then it will generate an error message.

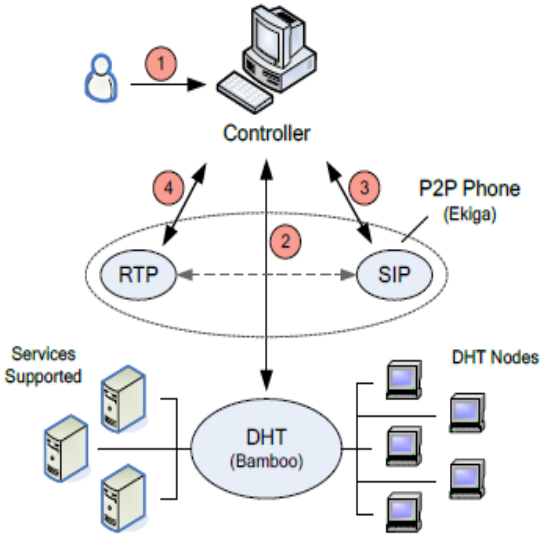

Figure 4. Functional Components of Our testbed

We use Bamboo [7] for the underlying DHT in which a node maintains two sets of neighbors, the leaf set and the routing table. Message routing may be performed recursively or iteratively according to these neighbor sets, and lookup is done in O(log N) hops. Compared with other DHTs, Bamboo can better withstand large membership changes as well as continuous churn in membership with the smallest delay. Its stability without bandwidth limitation is well suited for the real-time VoIP communications. Consequently, our toll suit uses Bamboo as the underlying DHT for looking up objects and routing messages.

Social DHT incorporates all the basic social network infrastructures, and it is the pivot of our society-integrated P2P VoIP system. Therefore, we introduce a new Social stage for the integration of social functionalities into social DHT. The Social stage maintains the mapping from Social ID to Node ID, and provides the social routing policy for the Router stage. Social stage is equipped with the fundamental function of the running of social DHT, and a good portability makes it compatible with any kind of P2P DHT.

The Experimental Setup consists of laptops running Linux Fedora Core 9 or 10 with the soft phones. The DHT has been redesigned to route calls based on the social call graph analysis as described above. Most of the Laptops have cameras to enhance the soft phones capability to transmit video over RTP. The setup uses Standard RTP protocol for transfer of media packets and uses SIP protocol for session initialization. The setup runs the phone platform of Ekiga 2.0.2 with its supporting packages - OPAL (Open Phone Abstraction Library) for the RTP message creation and QoS and LibPT which delivers the interfaces for audio and video to the hardware level.

We use a traffic generator to create 500 nodes and assigned random call traffic of around 40-100 contacts for every node. As the DHT scales the network to a maximum of 10,000 nodes, the nodes were created to share and exist evenly over the clusters in the network. The number of peers joining the DHT was monitored using the Vis tool. The Vis tool is a Visualizer which displays the peer's status on the network, its location and path etc. The initial experiment on the DHT was to scale the network to occupy a total of 500 nodes and achieve communication between peer entities. Our hardware could not withstand the load generated by the DHT where all the Laptops were distributed with 50 peers equally. With the addition of nodes joining the network, the performance of the machines degraded significantly because of the excessive flow of DHT update messages over the network. Because we limited the number of leaf-set neighbors to 6, our system didn't converge at any point of time. Then the experiment was done with the introduction of more number of leaf set entries for every node. The size of the messages and the packets increased drastically with a large number of leafs nodes and crashing the whole system. The network with the present hardware conditions support around 240 nodes on the network for a stable network. The Visualizer always indicated the total number of nodes as 220-260 even when all the 500 nodes are up and running.

Then the Experiment was modified to an optimal level of nodes for every laptop. The test bed was initialized with 100 nodes running the DHT overlay. The nodes with the leaf-set count still had the convergence problem as the Visualizer showed only 23 nodes were running, the measurements were taken using wireshark, a network sniffing tool to capture and analyze the packets over a network. The Primary results showed a delay for 10 to 25 seconds for the SIP messages. Hence the source was altered to have only 3 leafs and it has to maintain only 3 pairs of leafs which are closest and directly related to the node.

V. EXPERIMENTS

The Experiments were conducted using laptops with Ekiga 2.0.2 and a Video Codec of H.261 and Audio Codec of PCMU and a Jitter buffer of 20 ms. The test used the DHT overlay with all the 100 nodes running over the network. The measurements are made from the Ekiga debug engine from the statistic report sent from time to time between the two ends.

The first set of experiments was done to measure the performance of the system during initialization. The laptops were made to run only the required DHT overlay to get the accuracy closest to possible. The traditional Java time measurement is given by System.currentTimeMillis(), which uses the system clock to return the time to an Millisecond. Our experiment mostly returned a 0 ms time, since most of the code runs in sub millisecond values. Then we used the nanoTime() system call, which is introduced in the Java 1.5 version. The

system call uses the same wall clock (system clock) to return the value of time in nanoseconds. As per the literature, the accuracy of time will differ based on the architecture of the machine and the clock speed.

Some preliminary tests and measurements were made to determine the accuracy level of both the methods. The currentTimemillis() gave an accuracy of 1 to 2 ms. The same way nanoTime() method was tested and it gave an accuracy range of 1-1.3 microseconds, which is considered to be the best available time java can retrieve from the system. Both the methods were affected by the number of processes currently running in the system and the current payload being processed by the entire system.

The first test measured the registration time of nodes. The application or the node has to register to the DHT for the presence in the network and inform other peers for its contact. The Registration time is the time to interface the node to the DHT overlay. We first measured the registration time without social hashing and then we measured the times using social hashing. The results showed that for each node the registration time without social hashing was between 11ms and 15 ms. The registration time with social hashing was between 5ms to 8 ms.

The DHT identifies a node/peer using the GUID, at the initialization stage; the DHT assigns the GUID to the node at start up. Fig.5 shows the comparison of GUID assignment times where the X-Axis represents the node number and the Y –Axis represents time in microseconds. As the figure shows the GUID assignment times are significantly lower when social hashing is used.

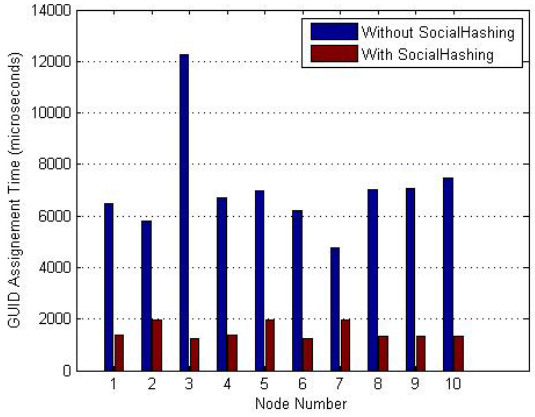

Figure 5: GUID Assignment Time

The Gateway assignment time is the time needed for the nodes to connect to their gateways based on the social information. Fig. 6 shows a comparison of Gateway assignment times for each node where the X Axis represents the node number and the Y –Axis represents time in microseconds. Again the table shows that assignment times using social hashing are significantly lower than the times when no social hashing is used.

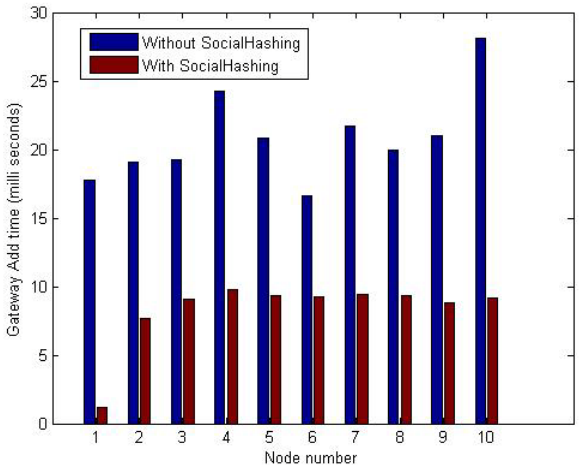

Figure 6: Gateway Assignment Time

The Social table has to be loaded for clustering and analysis, and the time taken will vary based on the number of social contacts for the peer. Fig. 7 shows the time taken to create social table, where the X-Axis represents the node number and the Y –Axis represents time in microseconds. This is an overhead for the social hashing scheme.

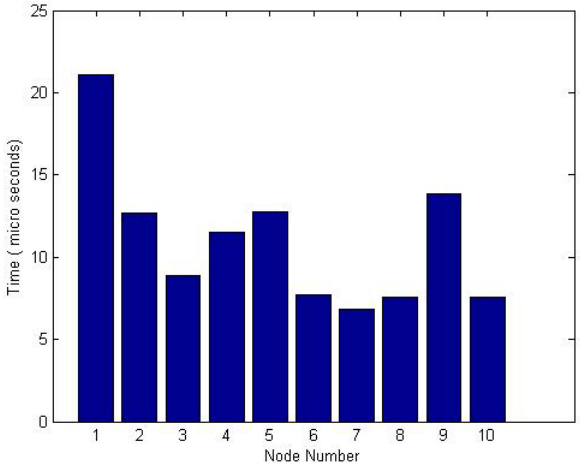

Figure 7: Social Table Creation Time

The fig.7 shows the times for different nodes, where the time taken for social table varies, because the number of contacts of a node has direct impact on the social table size and its creation time. We can observe that the number of contacts for the first node is the highest and the time corresponds to the social table size.

The Social ID generation is based on social hashing, which will be initialized for contact by other systems. These social ID's are generated based on the clusters and the group of cluster the node belongs to. Hence it categorizes the node for which cluster it has to join and the time for assignment of the ID will depend on difficulty of Hash and distant from the rest of the cluster.

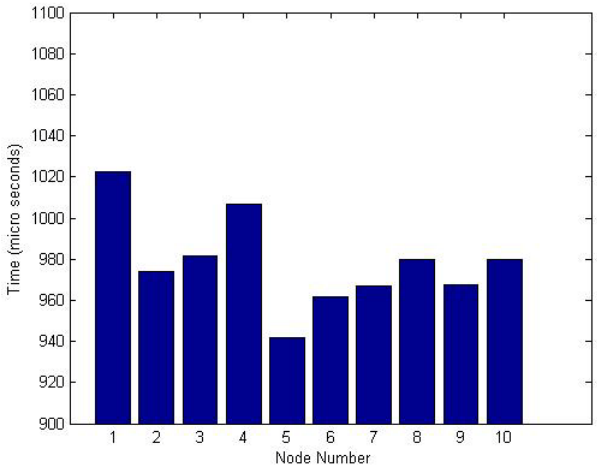

Figure 8: Social ID Assignment Time

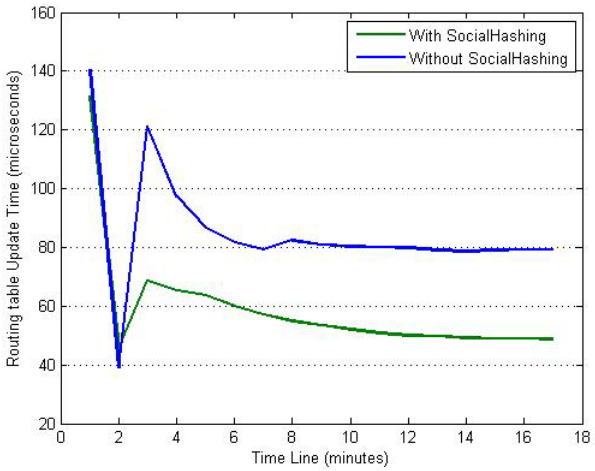

Figure 9: Routing Update Times

A Social ID is generated based on the hashing value information by the social hash function and this ID will be used for contact by the system. Fig. 8 shows these times for social hashing scheme, where X-Axis represents the node number and the Y –Axis represents time in microseconds. This is also an over head for the scheme. These operations are done only during initialization. So the overhead can be justified as the algorithm is much more efficient during run time and also during initialization of other common parameters like GUID assignment time etc.

Fig. 9 shows the Time taken for successive routing updates by the DHT and the time taken for calculating the routing changes and its update into the system. The Routing update time may differ based on the number of routing updates received in a time and the size of the routing update message. The results show the times for updates without using social hashing and also the times when social hashing is used. For both the schemes the results show that initially the routing update times are high, but as the system stabilizes, the times start to go down. The trend line for the routing times shows that over time the update times go down. But consistently the measurements taken for times using the social hashing are much lower than the case where social hashing is not used. Also the trend line for social hashing shows a steeper decline over time.

VI. Related Work

[10] Show how social networks can be used to derive trust in making routing decisions between Internet based services. In order to do so they propose SPROUT and DHT routing algorithm that uses social links. They show that by doing so increases the time it takes to answer routing related queries.

[11] Introduces Social VPNs in order to enable ad-hoc self configuring self managing and trust maintain social connections. The paper describes the architecture and a prototype implementation that integrates Face book API, IP-over-P2P virtual networks and IPSec. The paper show how their overlay network handles unmodified TCP/IP applications and NATs. A performance study is also presented in [11].

[12] Presents Davis Social Links (DSL) that integrates trust relationships present in social networks into the Internet communication infrastructure in order to provide receiver controllability, traceability and global connectivity in environments without centralized services or globally unique names. The objective of [11] is to provide defense against SPAM, SPIT, SPIM and DDoS attacks.

[13] Is a social science study that explores how directional and dyadic relations such as social relations can be attributed to the effect of the main actor, the secondary actors(s) and their relationship (kind). In order to do so, they use the Social Relation Model (SRM) [14] to analyze interpersonal relationships from multiple dimensions including homophile, identification and individual attraction.

VII. Conclusion

We have proposed using social network of telephone users as a basis for creating a P2P VoIP network. Based on this assumption, we have used a telecommunication user's social connections formed and nurtured using the telephone as a basis

to route calls. We have implemented this routing scheme by enhancing the Bamboo DHT algorithm and experimented with using soft phones installed on laptops as telephone customers as an initial prototype of our system. Our experiment, although need to be enhanced, showed reasonable delays compared to routing that does not use social connection as a basis for routing.

REFERENCES

[1] Xiaohui Yang; Dantu, R.; Wijesekera, D., "Achieving Peer-to-Peer Telecommunication Services through Social Hashing," Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE , vol., no., pp.1-2, 10-13 Jan. 2009.

[2] Xiaohui Yang; Dantu, R.; Wijesekera, D., " A Society-integrated Testbed Architecture for Peer-to-peer Telecommunications".

[3] Xiaohui Yang, "P2P VoIP Social Hashing Implementation Technical Report"

[4] Guido Urdaneta, Guillaume Pierre and Maarten van Steen. A Survey of DHT Security Techniques. ACM Computing Surveys, 2009.

[5] Hari Balakrishnan, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica. Looking up data in P2P systems. In Communications of the ACM, February 2003.

[6] S. Rhea, Handling Churn in a DHT. Proceedings of the USENIX Annual Technical Conference, 2004.

[7] The Bamboo Distributed Hash Table, available at http://bamboo-dht.org.

[8] The Ekiga Review, available at http://www.softpedia.com/reviews/linux /Ekiga-Review-29584.shtml

[9] The Vis Network Visualizer available at http://www.shareup.com/ Network_Visualizer -download-22270.html

[10] Sergio Marti, Prasanna Ganesan, Hector Garcia-Molina, SPROUT: P2P Routing with Social Networks, in EDBT Workshop, 2004, LNCS 3268, 2005.

[11] Renato J. Figueiredo, P. Pscar Boykin, Pierre St. Juste and David Wolinsky, Social VPNs: Integrating Overlay and Social Networks for Seamless P2P Networking, in Proceedings of the 2008 IEEE 17th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises - Volume 00, Pages 93-98, 2008.

[12] Lerone Banks, Shaozhi Ye, Yue Huang and S. Felix Wu, Davis, Social Links: Integrating Social Networks with Internet Routing, LSAD'07, August 27, 2007, Kyoto, Japan.

[13] Zuoming Wang and Joseph B. Walther, Interpersonal Perception in Virtual Groups: Examiing Homophily, Identification and Individual Attraction Using Social Relations Model, available at www.comm.unt.edu/facpres.htm.

[14] David A. Kinny, Interpersonal perception: A Social Relations Analysis, Guilford Publishers, New York. 1994.