# Filtering SPAM in P2PSIP Communities with Web of Trust

Juho Heikkilä and Andrei Gurtov

Helsinki Institute for Information Technology
Helsinki University of Technology
{Juho.Heikkila,Andrei.Gurtov}@hiit.fi
http://trustinet.hiit.fi/

**Abstract.** Spam is a dominant problem on email systems today. One of the reasons is the lack of infrastructure for security and trust. As Voice over IP (VoIP) communication becomes increasingly popular, proliferation of spam calls is only a matter of time. As SIP identity scheme is practically similar to email, those share the same threats. We utilized Host Identity Protocol (HIP) to provide basic security, such as end-to-end encryption. To provide call filtering, however, other tools are needed. In this paper, we suggest applying trust paths familiar from the PGP web of trust to prevent unwanted communication in P2PSIP communities.

The goal is to provide trust visibility beyond the first hop without requiring people to openly share private data such as contact lists. Since our distributed environment limits global solutions, our proposal bases on scale-free distributed nodes which provide service to the social trust neighborhood. We have implemented the service as a freely deployable stand-alone HTTP server, which can be either independent or a part of the P2P overlay. We have evaluated the performance of the path finding algorithm using the social network data from the PGP web of trust.

**Keywords:** p2p, social networking, trust, spam prevention.

## 1 Introduction

One of the most significant plagues of Internet today is spam, or unsolicited email. During the last ten years the amount of abusive email has grown from virtually zero to what is now estimated between 80 and 95 percent of email traffic [1][2] and to be in the order of 100 billion messages every 24 hours in June 2007 [3].

Today, spam filters are common in any aware environment, and they clear most of the garbage so users never see it. Still, some spam comes through and there are some false positives, not to mention the wasted bandwidth. Therefore the spam problem is not cured, but still remains, kept in the background with band-aids to ease the pain.

Thanks to broadband and mobile access, people are staying connected online more and more. As high-speed wireless and flat-rate tariffs spread, another form

A.U. Schmidt and S. Lian (Eds.): MobiSec 2009, LNICST 17, pp. 110–121, 2009.

of communication is becoming popular, Voice over IP (VoIP) calls, or Internet Telephony. SIP, Skype, Messenger, and other services provide people with ability to talk to each other in voice or video for practically no added cost. This opens new opportunities to the spam industry. Traditionally calling people has involved direct costs per second, but if soon more and more people are reachable without such direct costs, the aspect of building automated recorded calling services to advertise things like Viagra and adult services can become plausible and profitable.

These unsolicited calls, nick named SPIT (Spam over IP Telephony), pose an even greater harassment to people than traditional unsolicited email (SPAM) does. There are two fundamental differences between these forms of communication. First, while emails go to the inbox and wait until the user reads them, a call is made in real time and needs acute attention, disturbing anything the user is doing at the moment. Secondly, the content of an email message in the inbox is usually readily readable by the server running the spam filter. The content of a call is not available until the call actually happens, so it cannot be prefiltered based on content until the call is passed and answered. Therefore, voice calls are much more intrusive and they cannot be filtered based on content[1].

Methods for preventing SPIT need to be developed before it becomes a problem, otherwise we end up in a similar situation as with email, trying to fix things already broken. We can see that if VoIP systems are not designed to prevent SPIT, it will likely become the next pervasive medium for spammers [4]. In the worst case for VoIP, this means people will stray away from the medium, or there will be an abundance of incompatible non-standard systems, none of which is common enough to draw spammer attention.

Even in Finland, where unsolicited advertising is rare and prohibited, there has been talk of cases where automated services have called traditional phones asking for permission to make advertisement calls. As such calls are made in traditional networks, we can assume they will become common in their inexpensive counterpart (even if, as with SPAM, we expect the majority to come from global actors).

In this paper we will look at a distributed SIP environment and using PGP-like grass roots approach of a social web of trust to filter out calls coming from outside or too far in the social neighborhood.

The rest of the paper is divided as follows. Section 2 gives an overview of our underlying P2PSIP system and issues of trust in distributed systems. Section 3 outlines the research problem of preventing SPIT. Section 4 describes the proposed solution on a conceptual level, and Section 5 looks deeper into the actual implementation. Section 6 presents the test data and performance results and Section 7 concludes the paper.

## 2 A P2P SIP System

Session Initiation Protocol (SIP) [5], is a text-based IETF protocol standard for establishing and managing multimedia sessions. SIP handles tasks such as

[1] It may be noteworthy that voice-mail counters these two arguments, yet filtering audio based content is much harder than for text.

negotiating connection parameters and media encoding. SIP User Agents (UA), are identified by a SIP Address of Record (AOR), an email-address-like identifier.

In traditional, non-distributed, SIP, the user agents (UAs) can connect directly to each other if they know the connection parameters for the other party. In practice, however, a set of various trusted servers, such as SIP proxies and SIP registrars, is required for smooth operation of the service. Since this model has created administrative burdens, points of failure and a lack of ad-hoc capability, several P2P SIP systems have emerged and the IETF is currently standardizing the data protocol in the P2PSIP working group [6].

The Networking Research Group at Helsinki Institute for Information Technology (HIIT) has produced a prototype for such a decentralized SIP system [7], which is built on top of Host Identity Protocol (HIP) [8]. HIP communication architecture provides the tools required for achieving the identifier-locator split, i.e., separates the identity of a host from its location, such as the IP address. HIP hosts are identified by a Host Identifier (HI), the public key of a cryptographically generated key-pair. These are in turn dynamically mapped to network locations, providing transparent multihoming and mobility in addition to confidentiality and self-authentication. Therefore, we can use HIP to provide more secure identities and confidentiality to used channels, than SIP alone can. While HIP won't completely remove bootstrapping risks in distributed environment, it provides secure connections to verified parties. The HIP implementation used in the prototype is HIPL [9], which is available for mobile Linux platforms, such as the Nokia Internet Tablets.

Figure 1 presents the architecture. The P2PSIP peers are the devices of end users. These contain the user application (or UA) and also a local P2PSIP proxy. Instead of using large centralized proxies, the functionality is distributed to small lightweight proxies which share the workload. Moving proxy functionality from network servers to end devices is the basic concept behind P2PSIP systems. Various P2PSIP designs vary in which parts of the system they focus on, but

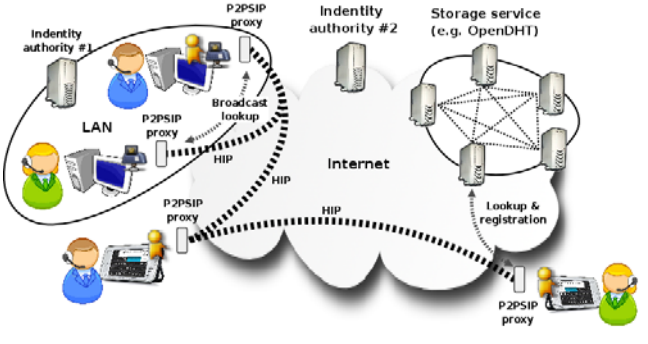


**Fig. 1.** The components of the P2PSIP system. In addition to users and peers, identity authority and overlay infrastructure can be present. The P2PSIP proxies handling the SPIT prevention are located in each end device. Peer-to-peer connections are made over HIP.

the local proxies forming a P2P overlay are common. Since the same proxy interface is still available, it has been possible to design the system to support legacy SIP applications.

The P2PSIP overlay network is not required for the UAs to make ad-hoc connections to each other, but it provides a global storage and an index. These services are essential for a smooth functioning of the system. The peers locate each other by the registrations stored in the overlay. Our implementation is not bound to a specific overlay, but can use multiple in parallel.

Finally, the Identity Authorities are centralized components for signing up for the system. These are trusted authorities which bind the SIP AOR to a public key using digital certificates, thus preventing identity thefts and spoofing. While Identity Authorities are centralized, there is no limit to their number, each providing service to their own domain, and once the registration and certification is done, there is no further need to stay in contact with the authority.

### 2.1 Trust in a Distributed System

We want to see how social trust relations could be used to filter calls. People seen as trustworthy by others in the social neighborhood should get through, while those seen in a negative light would be blocked.

Since our prototype is built on HIP and distributed SIP, the environment is by design distributed. This brings in new challenges, as we cannot rely on any centralized servers maintaining all the trust relations or reputation data. Balancing issues of privacy and reputation in a decentralized network is always a task requiring compromises, but our goal is to make the pay-offs greater than the costs.

In general, the question is a traditional introductory problem. If Ann wants to get in contact with Bob, but there is no direct link between them, Carol, who is acquainted with both Ann and Bob, can give Ann a reference to make her appear more trustworthy to Bob. However, in most cases we don't want to bother Carol explicitly, but only make it apparent to Bob that she sees Ann in a positive light. In more distant cases, where the trust chain is longer than two hops, a single introducer will not suffice. Further, unless Ann knows that Bob trusts Carol, she would be blind about knowing whom to ask for the recommendation. In a distributed network there is no node which would be guaranteed to know the best recommenders or shortest chains.

Even with limited knowledge we can make guesses and estimates. Without a comprehensive centralized database we cannot be certain we find the shortest or best link. If we do find a link, we will know that there is at least some link. While additional information could provide more trust, even a single link provides a basis for trust estimation. In case of purely positive reputation, it is also a guaranteed bottom estimate. Although we don't know if there are more available trust paths, we know those could only add to the estimate, not lessen it.

Social links can be used as credentials for contacting other people. People who are close in the social network are less likely to make SPIT calls. Just saying "Hi, I'm Ann, friend of Carol's" is likely to make Bob more receptive to talking

to Ann in a traditional telephone call. This can be contrasted to email spam, however, where just shooting random names is commonplace. To be actually useful in an online environment, there needs to be some more proof that the link is real and that the person on the other end is who she claims to be.

There is a limiting factor, however. People's contacts are private information, and most people are not willing to publicize their contact lists. Even when there is nothing explicit to hide, people do not wish their employer to know their personal friends, or all their contacts to have access to the contact information of their children. Privacy of contact lists must therefore be protected.

While counter-claims say that in the age of Facebook and MySpace people no longer demand such level of privacy, it is fair to note that social networking sites are voluntary. People can choose which of their real links they reveal, as well as whether their friend list is available to the public, their friends, or nobody. In contrast to mandatory publication of telephone contact list by an operator, many of the most open-minded social network users would protest revealing such information. The ability to control information about oneself is one of the central concepts in privacy [10]. This control is not about enforcing fixed rules, but about managing and balancing based on the environment [11].

## 3 Beyond the First Hop

With the need to protect private data, how do we enable people to see the social network beyond the first hop (their own contacts). Although asking the contacts to share their contacts could be plausible in some limited cases, it is not really a solution which could be widely accepted. Also, adding hops in such a manner as to pass on other people's contact lists to achieve deeper visibility, would be a breach of trust and privacy.

Likewise, sharing the contact lists with a trusted centralized agent could solve the problem. However, such a solution is not applicable to distributed environment and further, would build a gigantic database of global contacts. The latter, while not commonly seen as a major problem today, can become such after the first major abuse of such database presents itself.

People should be able to choose who has access to their contact lists, and also, to which part of it. Naturally, managing contacts to hide or reveal details is an added burden for the user. However, it should be welcomed by anyone with even slight privacy concerns.

Further, since the goal is only to present credentials for the caller, there is no need to reveal the plain identities of the intermediaries. Therefore, obfuscating the identities should be a standard practice when passing contact lists to other parties. After all, the plain names have no meaning for the social trust structure, those who's real identity would provide any additional value are likely known and on one's own contact list to begin with.

Finally, making iterative searches over a distributed network is a time-consuming effort. If one needs to query data from various nodes, and especially nodes more than one hop away, the process is likely to take too long to process for an incoming real-time call. Making such searches will also burden the

recipient node, (at least its network connection, which may be limited) making it vulnerable to Denial of Service (DOS) attacks.

We will present a method which provides trust information for nodes beyond one's own contact list, obfuscates user identities, hide identities of intermediary nodes to protect their privacy. The method places majority of the burden on the caller and on additional nodes which performance is not critical to the basic functionality of P2P SIP system.

## 4 Trusted Pathfinder Service

A Trusted Pathfinder is a service node in a flat network, which acts as a trusted third party to provide the call recipient with an anonymous trust path description of the caller. It fulfills this role by maintaining a protected and anonymized database of contact lists of people and finding trust paths in this web of trust. In a way it is similar to the web sites that find trust paths between PGP [12] keys, such as the PGP pathfinder [13] hosted by Utrech University.

To protect the privacy of the users, we have designed two obfuscating steps to minimize the exposure of private data. The first step is to hash all identities using SHA-1 at the client proxy. This will obfuscate the contents of the list even from the pathfinder service, preventing accidental exposure of administrators and requiring malicious admins to put at least some effort to decoding the identities. Hashing must not be confused with security, however. Secondly, the pathfinder will not reveal the intermediary nodes along the path, nor share the trust link data with any other parties. To preserve privacy of both private and business users, pathfinders must not share the contact data with each other.

The trust toward the Pathfinder is two-fold. First, the user who submits her (hashed) contact data to the service, should trust the service to keep her data private. Secondly, the user trusts that the path descriptions provided by the service are accurate to the extent possible with known data.

Note that both of these issues are in the human trust domain. This means that such issues can not be enforced by technical means, especially not in a peer-to-peer network. There is nothing to prevent a node administrator from running a service which appears legitimate, but leaks information. Submitting private data to a service ran by unknown actors is a risk and needs to be acknowledged as such. The P2P environment emphasizes the responsibility and decision-making of the user. Despite people's wishes, trust cannot be outsourced and there is no substitute for common sense and familiarity with the environment.

Our environment is distributed, so the pathfinder is not a global service either, but rather we see numerous local (in trust domain) services, and that the size distribution of the pathfinder nodes is likely scale-free [14]. Basically anyone could run their own pathfinder service; while the smallest ones would not have much public usage, it might be likely that circles of friends could run them, a company could run one for internal use, universities could run their own, etc. With client software modifications the user could further manage which contacts are shared with which pathfinder.

Since the pathfinders do not share their database with each other, a path query can be performed fast within the node without propagation delays. However, since the node does not contain global knowledge, the trust description it provides is limited. While this may sound like a limiting restriction, it may also model the trust network of people in a more human way. There is no need to be able to form paths with six degrees of separation, to every human on the planet. The goal is to reach beyond the first hop, the users' own contact lists. Since trust is not transitive, paths longer than three hops likely provide little introductory value. While the existence of any path could be interpreted as hint that the caller might not be a SPITter, infiltrating social networks is in practice easy as people find new contacts and adds only one to the length of the path. Adding a second 'cover' node adds two hops.

In the case of one additional hop, the friends of an infiltrating id could statistically find out who is responsible. If the path is longer, the question who has a friend with a spitting friend is very difficult to resolve without breaching the privacy of contact lists. Yet, the pathfinder itself, having internal access to the whole graph, could be modified to also run algorithms to locate regions linked by a single vertex, and consider those potential Sybil attackers similar to [15]. This functionality would be mostly useful in very large databases, however.

The pathfinder also acts as a small buffer against DoS attacks against the recipient. Since now the recipient proxy can require a path description before passing the call to the actual SIP client, the burden of pathfinding is on an external node. Certainly, the pathfinder can be taken down by an attack. However, since it is not a part of the P2PSIP infrastructure itself, but rather an additional service, removing it does not hinder the VoIP service. People who are one hop away (on a local contact list) can call freely without the pathfinder (though there may be other rules blocking the call). People can choose to accept calls without a path statement if they want. Attacking the pathfinder will, however, make life harder for those who are far, or not at all, in the social network, such as SPIT callers. Since direct contacts do not need the service at all, all delays posed by the system only apply to strangers.

Since the P2PSIP proxy is requiring the (unknown) caller to query the pathfinder service, all the delay in making the call can be put on the caller. Although, as we will show, the delay is not very long, it could be if the pathfinder is busy. Differentiating this waiting from a ringing tone should help a (human) caller perceive the call progressing. The recipient is not aware of this delay, his client starts ringing only after the call is passed through the proxy.

## 5 Implementation of the Pathfinder

We implemented the pathfinder as a stand-alone HTTP server written in Python [16]. The first goal was to test the performance need for tracing routes in the social network. A deployed call filtering service is likely to receive much higher load than the PGP pathfinder services on the web. While this is not an issue for small local servers, any nodes servicing a larger population may run into trouble if the workload is too high.

While we wanted a working solution and a proof-of-concept implementation to test, we didn't focus on optimization of the solution at this stage. There is likely still room for improvement in this regard for future versions of the implementation.

For the search algorithm we decided to keep to a basic two-way search [17], instead of putting an effort to tune a more complex algorithm. To optimize the search we built two data structures from the contact lists. The first contains the forward links as per the contact list. The other set contains reverse links readily available for the backward search.

The current implementation of the pathfinder provides the shortest trust path from the recipient of the call to the caller. This keeps the algorithm simple, as we do not need to keep track of the path or its properties. In future work we will consider adding metadata such as trust-weights to the links. Without such reservations for metadata we could have used Bloom Filters [18] for more secure obfuscation.

To begin, we add the recipient to a forward horizon, a list of nodes seen at a particular distance from the recipient, and mark the distance as zero. Then we add the caller to a reverse horizon and check if it matches the opposite horizon. If yes, we return the distance. As the caller likely is not the recipient, we increase the distance counter and return again to the other (forward) side. Then we start building a new horizon from unseen nodes on the contact lists of nodes on the previous, lower, horizon, and check if any of these match members of the opposing horizon. To prevent loops and to optimize the algorithm, we keep a record of nodes already seen, so we can ignore them in future. This process is iterated until the horizons meet, the maximum path length is reached, or all nodes are visited.

The algorithm, in a step by step manner:

- Take contacts from contact lists of a member of old horizon
- If these have not yet been seen, add to a new horizon
- If the new member matches the opposite horizon, return distance
- If the new horizon is completed and no match is found, increase the distance counter and iterate on the opposite side

The server also fits a contact list parser module, which interprets the contact list format(s) provided by user proxies to the database format used by the pathfinder. The current implementation supports our own XML schema, but a deployed implementation benefits from ability to understand various formats. As the data is never supposed to be exported, only import functionality is needed.

Finally, to provide integrity to the path description documents, a digital signature needs to be added to the response. For this purpose we add an S/MIME compliant signature to the document returned by the pathfinder server as a result of the query.

Figure 2 shows the modules in the implemented system as well as the processing paths for submitting contact lists and making path queries.

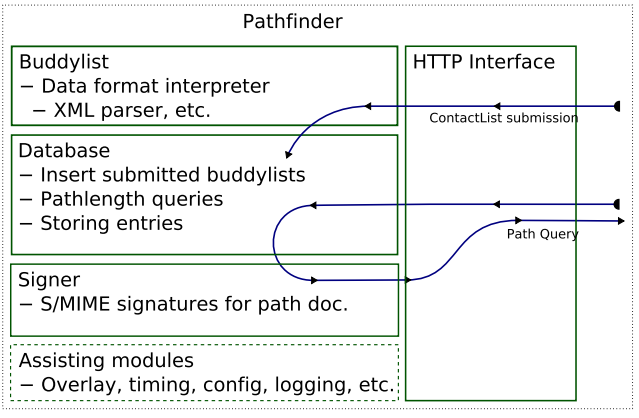


**Fig. 2.** The modules in the pathfinder system. The HTTP server accesses services provided by the database, XML-parser and signer, in addition to small help modules.

## 6 Performance

The pathfinder is now functional and can be used in conjunction with the P2PSIP system [19]. Although some performance issues remain, the median performance is suitable for small to medium scale deployment. We have a few optimization and scalability plans for the future, even as the current performance with the test dataset is satisfactory.

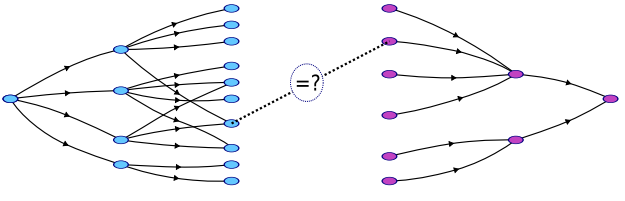


**Fig. 3.** The horizons in the social network are browsed by turns until they meet

### 6.1 Wotsap PGP Signature Data-Set

As a test dataset for the social network, we chose to use the Wotsap PGP key signature dataset available from the Wotsap website [20].

The dataset differs slightly from a real-world contact database since it only contains a strongly connected set of keys. In other words, each key in the set is reachable from every key in the set. There are no isolated nodes, or ones that have only incoming or outgoing links, unless we decide to insert them ourselves.

The links in the data-set are unidirectional, making it in similar to generic contact lists. Here, the topology differs from many social networks such as Facebook [21] or LinkedIn [22], but we feel that this approach is a better model for trust issues as trust is not generally symmetric.

The dataset contains about 35 000 keys and 350 000 signatures, giving an average of about 10 links and median of 3 links per node. While it can be argued that a realistic social network would be more dense, after consideration we decided that the dataset represents a generic social network in a sufficient manner for testing purposes.

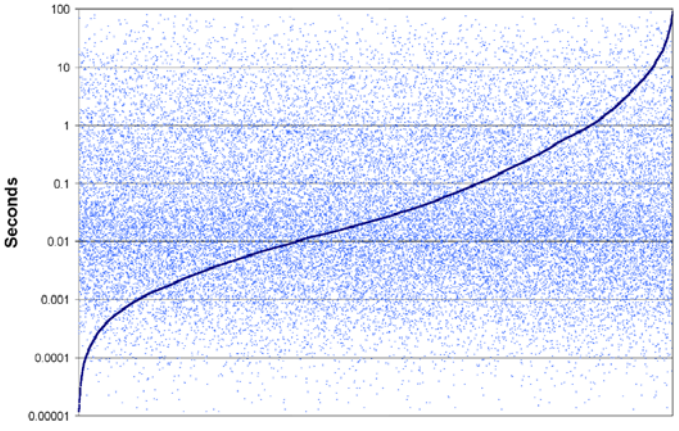


**Fig. 4.** The performance of pathfinder. Times of finding a path from a random node to a random node. 30 000 iterations shown in test order (cloud) and sorted by time (curve).

### 6.2 Measurements

We fetched 30 000 random-to-random paths using the pathfinder, setting the maximum path length at 6 hops. For real use such trust settings could be too long, but in the spirit of the six degrees of separation theory we wanted to see results for even longer chains. As noted above, our network is sparse, so not every chain fitted into these 6 hops. The measurements were made on an Athlon XP 3500+ single core PC.

The median time to find a path is 23 ms, which is quite applicable for lightly loaded servers, but of course insufficient for heavy loaded servers processing hundreds of requests per second. The deviation of the results is quite wide. While 10% of the cases take less than 1 ms, the highest 10% take more than 1.9 seconds to a maximum of 104 seconds.

In over 90% of cases the added delay is acceptable, but when the delay builds to the class of 3-5 seconds and more, it is likely irritable to a caller. Especially when considering possible HIP base exchange delays on low-power devices, as well as other network delays, informing the user on call set-up progress needs to be addressed in the design of the user interface.

The delay is not directly connected to the social distance, but the work done by the server. This can be seen from Figure 5 where the correlation between the path search delay and the number of unique nodes encountered in the process

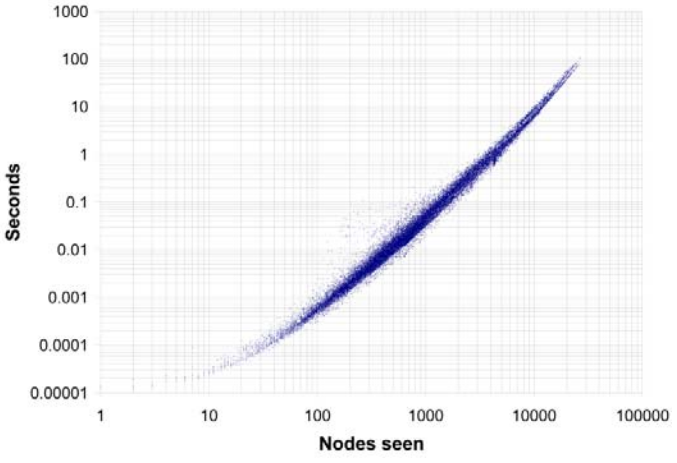


**Fig. 5.** The performance of pathfinder. Here the path finding time is presented in relation to the number of nodes seen on a loglog scale.

is shown on a loglog scale. We would also like to point that people who share regular contact with each other are likely be on each others' contact lists and thus should not be required to fetch path credentials when calling. Path search is a kind of introducer service, and the delay is an additional credential fetching cost which only applies to callers who are not on recipient's contact list.

## 7 Conclusion

We have designed and implemented a privacy preserving service for finding social trust paths between nodes which are further than one hop away from each other in the P2PSIP social network. Users submit their obfuscated contact lists to the trusted pathfinder service, which at request provides digitally signed credentials for the caller to present to the call recipient proxy. The pathfinders will keep the contents of user contact lists private, and will not reveal the path itself, only a description of it (path length in the current implementation). Large pathfinders could also try to locate likely Sybil attackers. We ran performance tests on the implementation and found it to be adequate.

## References

1. Messaging Anti-Abuse Working Group: Email metrics program: the network operator's perspective, report #7 (2008)
2. spamhaus.org: Effective spam filtering, `http://www.spamhaus.org/effective_filtering.html` (Referenced: 2008-08-06)
3. spamunit.com: Spam Statistics, `http://www.spamunit.com/spam-statistics/` (Referenced: 2008-08-06)

4. Croft, N.J., Olivier, M.S.: A Model for Spam Prevention in IP Telephony Networks using Anonymous Verifying Authorities, University of Pretoria (2005)
5. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: SIP: Session Initiation Protocol, RFC 3261 (Proposed Standard) (2002)
6. IETF P2PSIP WG, http://www.ietf.org/html.charters/p2psip-charter.html
7. Koskela, J.: A HIP-based peer-to-peer communication system. In: Proceedings of the 15th International Conference on Telecommunications (2008)
8. Moskowitz, R., Nikander, P.: Host Identity Protocol (HIP) Architecture, RFC 4423, Informational (2006)
9. HIP for Linux, http://hipl.hiit.fi (Referenced: 2008-08-06)
10. Altman, I.: The environment and social behaviour: Privacy, personal space, territory, crowding, Brooks/Cole Pub. Co. (1975)
11. Palen, L., Dourish, P.: Unpacking privacy for a networked world. CHI Letters 5(1) (2003)
12. Zimmermann, P.: PGP User's Guide, Volume I: Essential topics (1994), http://www.pa.msu.edu/reference/pgpdoc1.html
13. Penning, H.P., Feisthammel, P.: PGP pathfinder and statistics, http://pgp.cs.uu.nl/ (Referenced: 2008-08-25)
14. Barabasi, A.-L.: Linked: How Everything Is Connected to Everything Else and What It Means for Business, Science, and Everyday Life, Plume Books (2003)
15. Yu, H., Kaminsky, P., Gibbons, P., Flaxman, A.: SybilGuard: Defending Against Sybil Attacks via Social Networks. In: Proceedings of the ACM SIGCOMM 2006 (2006)
16. http://www.python.org/
17. Aura, T.: Fast access control decisions from delegation certificate databases. In: Boyd, C., Dawson, E. (eds.) ACISP 1998. LNCS, vol. 1438, p. 284. Springer, Heidelberg (1998)
18. Wikipedia, Bloom filter, http://en.wikipedia.org/wiki/Bloom_filter (Referenced: 2008-08-14)
19. Koskela, J., Heikkila, J., Gurtov, A.: A secure P2P SIP system with SPAM prevention. In: Poster in Mobicom 2008 (2008)
20. http://www.lysator.liu.se/~jc/wotsap/ (Referenced: 2008-08-14)
21. http://www.facebook.com/
22. http://www.linkedin.com/