Distributed Anonymous Computation of Social Distance

Edoardo Biagioni, esb@hawaii.edu
Department of Information and Computer Sciences, University of Hawaii at Mānoa, Honolulu, Hawaii, USA

***Abstract*—In a distributed social network, no single system holds information about all the individuals in the network, and no single system is trusted by all the individuals in the network. It is nonetheless desirable to reliably compute the social distance among individuals. This must be done anonymously, without giving away any identifying information about individuals in the social network, and reliably, without allowing anyone to pretend to be socially closer to someone else than they actually are.**

The Social Network Connectivity Algorithm, or SoNCA, accomplishes these goals in a distributed manner. This paper describes both the high-level algorithm and a concrete design that is intended for future use with a network, AllNet, designed to provide secure interpersonal communication utilizing all available means, including Internet, cellular communications, ad-hoc networking and delay-tolerant networking.

***Index Terms*—social distance; social network; ad-hoc network; peer-to-peer network; p2p; allnet**

I. Introduction

It has been suggested since the 1920s [1] and known since the work of Milgram ([2], summarized by Kleinberg [3]) in the 1960s that modern human societies are closely connected, leading to the proverbial "six degrees of separation". More rigorous work has found even closer connections in the age of social networks [4].

If the world population is at most six degrees of separation, and each individual has a description of their own social network up to three degrees of separation (friend of friend of friend, or f^3), individuals comparing such social networks are likely to find at least one match. If, as more recent studies show, the average degree of separation is closer to four, then comparing just two degrees of separation might often lead to finding a friend of friend (f^2) in common. Either way, recording about a million people on each device makes it very likely that any two devices will have in common information about at least one person.

The AllNet project [5][6] has the goal of persuading people to help others communicate. Devices that participate in AllNet may forward messages to and from other devices. Such forwarding consumes resources, particularly storage, battery, and bandwidth. For me to persuade you to carry my messages, it would be helpful if I could convince you that we have common friends or friends-of-friends (other ways of persuading people are described in an earlier article [7]). The AllNet Social Network Connectivity Algorithm, or SoNCA, is designed to make such exchanges automatic and efficient. Although designed in the context of AllNet, the algorithm has wider applicability to any distributed social network where it might be beneficial to establish social distance or similar metrics.

This paper first presents (Section II) an overview of AllNet, the security model used in this paper, and the privacy this design is intended to maintain. The main section (Section III) explains how SoNCA maintains the desired privacy while providing the desired information about degrees of separation. The analysis section (Section IV) considers the efficiency and security of SoNCA.

II. Background

A. AllNet Design

The early Internet was optimized for a few driving applications, most notably telnet, ftp, and email. For AllNet, driving applications include secure chat, telephony, and internet access. The first two need not use any infrastructure if the mobile devices are in range of each other or have sufficient intermediate devices.

Just as the Internet carries IP traffic over many other networks such as Ethernet and 802.11, AllNet is an overlay network that can run on top of other networks. Unlike the Internet, addresses in AllNet are optional. When no specific route to a destination is known, AllNet uses broadcasts limited to a specific number of hops. Packets sent with a lower number of hops are given higher priority than similar packets sent with a higher number of hops, incentivizing senders to use the lowest number of hops that will accomplish their goals.

More details of message addressing and forwarding are given in earlier papers [5][6].

B. Security Model

As in most research in network security, we assume the end devices are secure. While this may not always be true, no security protocol can provide security to an insecure device.

We assume an attacker can eavesdrop on messages, inject new messages, or act as an attacker-in-the-middle.

We also assume that conventional public-key encryption and authentication algorithms such as RSA [8] are secure, and similarly for secret-key encryption such as AES [9] and secure hashes such as SHA512 [10] used for authentication as part of an HMAC.

All interpersonal messages in AllNet are encrypted and authenticated using public-key systems. For larger volumes of data, session keys are exchanged securely under the public-key cryptosystem. Broadcast messages are not encrypted, but may be authenticated.

TABLE I
SYMBOLS USED IN SECTION III

Symbol	Meaning
f, f^1	friends' IDs
f^i, $i > 1$	IDs of the friends of the f^{i-1} group
$\|f^i\|$	number of IDs in f^i
k, k_1	number of bits in an ID, typically 128
k_i	number of bits saved for IDs in f^i, typ. $k_{i-1} - 16$
b	number of bits in the Bloom filter
n	number of IDs (1 bits) in the Bloom filter
h	num. bits exchanged for each potentially matching hash

Making contact with a new person and exchanging keys in AllNet is decentralized, and somewhat resembles the mechanism in `ssh`. This is different from the certificate mechanism used in SSL/TLS [11]. A certificate authority, may have little knowledge of the websites it certifies, but its certificates are widely accepted. In AllNet I certifies to myself only my own personal contacts. While a compromise of a TLS certificate authority might be used to impersonate many different websites [12], a compromise of Alice's keys might be used to send messages in Alice's name, but cannot be leveraged to impersonate Alice's friend Bob.

We assume that devices can generate random bitstrings, and that a random bitstring, if long enough, is unique with sufficiently high probability[1].

C. Privacy and Social Networks

We wish to have two individuals exchange sufficient information to establish their distance in the social network, without requiring any mutual trust or exchanging any secrets. Each individual begins the exchange with information about a small part of the entire social network, about 1 million people. The only information that each party can derive from the exchange is the path through their portion of the social network through which the other individual is reachable.

III. SOCIAL NETWORK CONNECTIVITY

The Social-Network Connectivity Algorithm (SoNCA) is designed to be used as part of a negotiation where one party (Alice) wishes to find out and have evidence of the shortest social network path between herself and another individual (myself).

Because no digital signatures are used in this process, individuals are identified by unique, self-selected, fixed-length (k-bit), persistent random bitstrings. Selecting $k = 128$ makes the chance of collision vanishingly small.

A. AllNet Social Network Connectivity Algorithm (SoNCA)

My device stores $|f|$ friends' IDs. Each ID has $k = k_1$ bits. Each of my friends also gives me the first $k_2 < k$ bits of all of their friends' IDs, and $k_3 < k_2$ bits of the IDs of their friends' friends. These IDs are collected directly from our immediate contacts, and kept on our devices. If $k = 128$, we can use $k_2 = 112$ and $k_3 = 96$.

If everyone has $|f| = 100$ random $f$riends and contacts, $|f^2| \approx$ 10,000, $|f^3| \approx$ 1M, $|f^4| \approx$ 100M, and so on.

The world is not random, so a reasonable goal is to limit the number of IDs stored, for example to about 10^6 (1 million) IDs. If the total number of IDs is 10^{10} (10 billion), the probability of *one* of my IDs matching one of someone else's 10^6 IDs is 1/10,000 ($10^6/10^{10}$), so my million IDs will average 100 matches. Therefore, for anyone on the planet, I will likely be able to determine the distance of their relationship to me, even though my device only stores a million IDs.

At 96 bits for most of the IDs (since most of the IDs are in f^3), SoNCA requires about 12MB of storage. Assuming $10,000$ people in f^2, every pair of contacts exchanges about $140KB$.

1) Proving Connectivity: Alice and I have never met. We have no way to identify each other, but I would like to discover, and prove to her, how far apart we are in the social network. Because we have no way to identify each other, the exchange takes place entirely in the clear. At the end of the exchange, Alice and I have only learned how closely connected we are.

I do not wish to send in the clear either my own or anyone else's ID, since such information could be used to impersonate me or my friends. However, if Alice sends me a nonce, I concatenate the nonce to each of the IDs that I have, and send Alice both the nonce and the hashes of the result. If Alice performs the same operation, she can tell whether we have any IDs in common. Assuming it is hard to reconstruct a bitstring given its hash, Alice (and any attacker) cannot reconstruct my IDs, but only confirm whether she has them already.

Computing and exchanging a million 512-bit hashes can take significant time and energy. To optimize this, I begin the exchange by sending Alice a Bloom filter reflecting the first few bits of the IDs that I have. In practice, a 16-million-bit Bloom filter can be built with the first 24 bits of the IDs (the optimal size of the Bloom filter is analyzed in Section III-B). Possession of these 24 bits is not particularly helpful to an attacker trying to reconstruct 96-bit or longer IDs.

If any of the bits in the Bloom filters corresponds to any of Alice's IDs, Alice has tentatively[2] identified one or more IDs that might match one or more of my $f^i | i \leq 3$. She then sends me the information about which bits in the bloom filter matched, and I send her only the corresponding hashes. Alice can then find the matching hashes in her f^3, f^2, or f^1, and compute my connectivity accordingly. By hashing a matching ID with a different nonce, Alice can then prove the same connectivity to me.

The exchange is repeated three times, beginning with a bloom filter of the hashes in my f, then f^2, then f^3.

If I don't want to reveal to Alice the exact size of my f^i set, I can exchange only a subset of my IDs, or create new random IDs and temporarily treat them as if they were real IDs, or both.

[1] If two people ever choose the same ID, someone may be able to claim a closer relationship than is actually the case, but nothing more.

[2] While a Bloom filter has no false negatives, it can have false positives.

2) Nonce Selection: Because I know nothing about Alice when she first contacts me, there is no way I can validate anything about her. I may not even see her in person – she might be within wireless range, but in a different room. All I know is the information that Alice sends me about the contacts we have in common.

This lack of knowledge allows an attacker-in-the-middle between me and Alice to perform validation, pretending to be Alice when communicating with me, and pretending to be me when communicating with Alice. The only thing the attacker cannot do is change the nonce. The nonce then has to be selected to be valuable to the person requesting the service. The nonce should then include the public key that Alice will later use to communicate with me.

B. Minimizing the Size of the Exchange

The SoNCA exchange is designed primarily to be used when devices are either directly in range of each other, or both connected to the Internet. In these cases, the number of bits exchanged is not likely to be critical. However, it is interesting to consider how the SoNCA exchange can be optimized by minimizing the number of bits sent and received.

When I first connect to Alice to initiate the SoNCA exchange, I must send her Bloom filters corresponding to my friends' IDs. For simplicity, this analysis only considers the set f^3, which in most cases will be much larger than the other sets, and has size $n = |f^3|$. The size of the Bloom filter for f^3 is $b > n$ bits. Assuming that IDs are uniformly distributed, each ID that Alice holds will match a bit in my Bloom filter with probability n/b. If Alice has $|f^i|$ IDs at level i, the expected number of matches between my Bloom filter and Alice's IDs at her level i is $n \times |f^i|/b$.

I wish to choose b to minimize the total number of bits exchanged, while still identifying every matching ID.

The number of bits I send to Alice initially is the number of bits b in the Bloom filter.

The information that Alice sends to me is the list of bits in my Bloom filter that match her IDs. In response, I will send Alice a hash for each match. If the total number of bits that Alice and I exchange for each possible match is h (the value for h is derived in the next two paragraphs), then the expected number of bits that Alice and I exchange for all the matches is h times the number of matches, or

$$h \times n \times |f^i|/b \tag{1}$$

To compute h, we note that the information that Alice sends to me for each match is the list of bits in the Bloom filter that match her IDs at level h. This can be encoded using $log_2 n$ bits for each match.

The hash that I send to Alice for each match can be a partial hash, since there is no practical purpose in sending more hash bits than the number of bits of the ID. If Alice stores k_i bits of the ID and I store k_j bits of the ID, then I need to send Alice $min(k_i, k_j)$ bits for each hash. The value of h in Equation 1 is then $h = log_2 n + min(k_i, k_j)$.

Including the original Bloom filter and the hashes, the expected number of bits Alice and I exchange is:

$$\hat{bits} = b + h \times n \times |f^i|/b \tag{2}$$

$|f^i|$ is unknown until the exchange is complete (we don't know which of Alice's levels, if any, will contain the ID of one of my friends), but if the number of stored IDs is limited to one million, $|f^i| \leq$ 1,000,000.

To find the optimal b to minimize the total number of bits sent, we differentiate equation 2 with respect to b and set the difference to 0.

$$\frac{d\hat{bits}}{db} = 1 - \frac{h \times n \times |f^i|}{b^2} = 0$$

$$b^2 = h \times n \times |f^i|$$

$$b = \sqrt{h \times n \times |f^i|} \tag{3}$$

Using $|f^i|$ = 1,000,000 I choose the nearest b that is a power of 2.

For example, if I have about 1,000,000 IDs in my f^3 and $k_3 = 96$ bits, h is $k_3 + \log_2 1,000,000 = 116$, so $b = \sqrt{116 \times 10^6 \times 10^6} = 11 \times 10^6$ or 11 million bits. The nearest power of two is 2^{24} bits or 2,097,152 bytes.

The likelihood of any one of Alice's 1,000,000 IDs matching a bit in my Bloom filter is about $1/16$, since $1/16$ of the bits in the Bloom filter are set. The expected number of matches is then 62,500. For each of these matches I must send a 96-bit (12-byte) hash, amounting to 750,000 bytes. The total amount of data I send is then 2,847,142 bytes.

That this is optimal can be verified by looking at the next lower power of two, using 2^{23} bits in my Bloom filter, which leads to an expected 125,000 matches and my sending Alice 2,861,076 bytes. The next higher power of two, 2^{25}, only has an expected 31,250 matches whose hashes can be sent with 453,125 bytes, but the Bloom filter has 4,194,304 bytes, so the total data I send is 4,647,429 bytes. Both of these are higher than the 2,847,142 bytes I must send when the Bloom filter has 2^{24} bits, so from this example we see that $b = \sqrt{h \times n \times |f^i|} \approx 2^{24}$ is indeed optimal.

IV. Security Analysis of SoNCA

When using SoNCA to establish a connection with Alice, I will send her some information about my social network. However, I know nothing about Alice, and in fact, she might be an attacker trying to obtain information from me. Since the exchange is not encrypted, any eavesdropper or man-in-the-middle attacker might also be able to get the same information. It is therefore appropriate to clearly identify what information is and is not sent.

No IDs are sent in SoNCA, only ID hashes.

As long as the hash functions are cryptographically secure, an attacker cannot identify the IDs in my social network unless the attacker has already obtained the same IDs.

The main concern is that an attacker who already has one of the IDs in my f^i will be able to tell that I have the same ID. While this might pose some concerns, it is also the point of SoNCA that someone I don't know should be able to tell what our degree of relationship is, and therefore that we have received the same ID from a person to which we are more or less closely connected. So it is necessary for Alice to be able to identify the IDs I hold and that she also has, and an attacker will be able to do the same.

There are ways to lessen the amount of information sent.

In variant A, I only send Alice the hash of my own public ID, giving an attacker a chance to figure out who I might be, but no information about my social network. Then, Alice will only be able to see if I am in her $f^1 \dots f^3$.

Variant B assumes that the information about IDs in my f^3 (my friends of friends of friends) holds little interest to an attacker, since I do not know these people myself. One of these IDs is very likely to be found in Alice's $f^1 \dots f^3$, and Alice may find that we are connected within her f^6.

Variant C combines variants A and B. First I send Alice the hash of my own ID, which Alice can use to figure out if I am in her $f^1 \dots f^3$. If I am not, I use variant B and Alice can see I am in her $f^4 \dots f^6$.

V. Related Work: Peer-to-Peer Social Networks

In recent years several proposals have been made for Peer-to-Peer (P2P) social networks. These include PeerSoN [13], Safebook [14], LibreSocial (`libresocial.com`) and PeerBook [15], though the last project no longer seems to be maintained. The very first secure P2P network was Phil Zimmerman's PGP [16], which provides security encryption for email messages. PGP introduced the idea of a web of trust based on social relationships. The major disadvantage of PGP is the practical difficulty of securely exchanging public keys, a difficulty that AllNet has addressed.

PeerSoN and Safebook appear to be progressing as good P2P social networks providing privacy and addressing many of the issues needed to provide a secure P2P social network. However, unlike AllNet, the focus of these project is on providing a distributed social network rather than a networking technology that will provide useful services whenever possible.

Freenet [17] is another network technology focused on secure data exchange among peers. Freenet is really designed for anonymity, and does not support the mutual exploration of social networks.

Work in reputation-based systems (e.g., [18], [7]), to reward socially constructive behavior and discourage socially damaging behavior, is complementary to SoNCA.

Delay-Tolerant Networks (DTNs) rely on the movement of devices, often carried by people. Some research has been done on using social network information to optimize delivery of messages [19].

VI. Summary and Conclusion

SoNCA allows two individuals, who do not trust each other, to determine the set of shared IDs in their respective social networks. Since randomly generated IDs are very likely unique, this gives high confidence in the determination of the shortest path connecting the two in the overall social network, and may lead to higher trust between the two.

SoNCA is not yet implemented in AllNet. AllNet does have the basics of peer-to-peer networking, including limited broadcasting of messages, keeping track of peers, bootstrapping, and prioritization and persistent storage of messages.

VII. Acknowledgements

Over the years many have expressed support and discussed with me the ideas in AllNet. Particular contributions are from Marifel Barbasa, Andreas Brauchli, Tiago Couto, Caterina Desiato, Henry Eck, Amanda Ishikawa, and Wes Peterson, and the taxpayers of Hawai'i who have supported this work.

References

[1] F. Karinthy, *Láncszemek (Chains)*, 1929.
[2] J. Travers and S. Milgram, "An experimental study of the small world problem," *Sociometry*, vol. 32, 1969.
[3] J. Kleinberg, "Complex networks and decentralized search algorithms," in *Proceedings of the International Congress of Mathematicians (ICM)*, 2006.
[4] L. Backstrom, P. Boldi, M. Rosa, J. Ugander, and S. Vigna, "Four degrees of separation," arXiv:1111.4570v3, 2012.
[5] E. Biagioni, "A ubiquitous, infrastructure-free network for interpersonal communication," in *The Fourth International Conference on Ubiquitous and Future Networks*, Phuket, Thailand, 2012.
[6] ——, "Ubiquitous interpersonal communication over ad-hoc networks and the internet," in *47th HICSS (Hawaii International Conference on Systems Sciences)*, Waikoloa, Hawaii, Jan. 2014.
[7] C. Desiato and E. Biagioni, "Sharing networking resources to create a pervasive infrastructure," in *Ninth International Conference on Technology, Knowledge, and Society*, Vancouver, Canada, 13-14 January 2013.
[8] R. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, February 1978.
[9] NIST, "Announcing the advanced encryption standard (aes)," http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf, US National Institute of Standards and Technology, Tech. Rep. Federal Information Processing Standards Publication 197, Nov. 2001.
[10] Information Technology Laboratory, "Secure hash standard (SHS)," FIPS PUB 180-3, Gaithersburg, MD 20899-8900 USA, June 2007.
[11] T. Dierks and E. Rescorla, "The transport layer security (tls) protocol version 1.2," RFC 5246, August 2008.
[12] "Hackers issue fake security certificates for CIA, Google," *Electronista*, Sep. 2011.
[13] S. Buchegger, D. Schiöberg, L. H. Vu, and A. Datta, "PeerSoN: P2P social networking - early experiences and insights," in *Proceedings of the Second ACM Workshop on Social Network Systems 2009*, Nürnberg, Germany, March 31, 2009, pp. 46–52.
[14] L. A. Cutillo, R. Molva, and T. Strufe, "Safebook: a privacy preserving online social network leveraging on real-life trust," *IEEE Communications Magazine*, vol. 47, no. 12, December 2009.
[15] B. Birt, "Peerbook," June 29 2010. [Online]. Available: http://blogs.cs.st-andrews.ac.uk/peerbook/
[16] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer, "OpenPGP message format," RFC 4880, November 2007. [Online]. Available: http://tools.ietf.org/rfc/rfc4880.txt
[17] I. Clarke, "A distributed decentralized information storage and retrieval system," 1999. [Online]. Available: http://freenetproject.org/papers/ddisrs.pdf
[18] S. Buchegger, "Coping with misbehavior in mobile ad-hoc networks," Ph.D. dissertation, EPFL, Lausanne, Switzerland, 2004.
[19] K. Chen and H. Shen, "Utilizing distributed social map for lightweight routing in DTNs," *IEEE/ACM Transactions on Networking*, no. 5, October 2014.