Ubiquitous Interpersonal Communication over Ad-Hoc Networks and the Internet

Edoardo Biagioni
February 2013

ABSTRACT

The hardware and low-level software in many mobile devices are capable of mobile-to-mobile communication, including ad-hoc mode for 802.11, Bluetooth, and cognitive radios.

We have started to leverage this capability to provide interpersonal communication both over infrastructure networks (the Internet), and over ad-hoc and delay-tolerant networks composed of the mobile devices themselves.

This network is fully decentralized so it can function without any infrastructure, but takes advantage of Internet connections when available. Devices may communicate whenever they are able to exchange packets. All interpersonal communication is encrypted and authenticated so packets may be carried by devices belonging to untrusted others.

One challenge in a fully decentralized network is routing. Our design uses Rendezvous Points (RPs) and Distributed Hash Tables (DHTs) for delivery over the Internet, and hop-limited broadcast and Delay Tolerant Networking (DTN) within the ad-hoc network.

Each device has a policy that determines how many packets may be forwarded, and a packet prioritization mechanism that favors packets likely to consume fewer network resources. A goal of this design and implementation is to provide useful interpersonal communications using at most 1% of any given resource on mobile devices.

Keywords

Infrastructureless Communication, Interpersonal Communication, Ad-Hoc Network, Delay-Tolerant Network, Networking Protocol, Priority Mechanism

1. INTRODUCTION

In the days of Plain Old Telephone Systems (POTS), a company would buy or rent an expensive PBX to connect the telephones in its offices to the worldwide telephone network.

Today many individuals own an inexpensive packet switching system to connect multiple computers to the worldwide Internet.

Some people have already used smartphones or other mobile devices as personal hotspots, obtaining Internet access without having to purchase a separate piece of equipment. At present, the performance of these hotspots is very limited, and the hotspot itself must have access to the Internet.

This paper is about using ad-hoc technology to extend the reach of personal hotspots both further from the Internet and to a wider group of people than just the owners of hotspots, to the point where people can communicate even without the Internet. The goal is to to provide low-bandwidth interpersonal communication and peer-to-peer social networking without regard to availability of infrastructure or ability or willingness to pay for a commercial service.

Ad-hoc technology is inefficient and unreliable compared to today's Internet. The advantage of ad-hoc and peer-to-peer technologies is that to work they only need two or more suitable general-purpose devices. The project described here, AllNet, is designed to take advantage of the Internet and other infrastructure when available, and use ad-hoc and delay-tolerant networking to continue delivering packets when Internet access is not available.

Any project that accomplishes these goals is likely to share these features:

- Support for mobile devices. Many people communicate using mobile devices. These devices are wireless and self-powered, and can be used (for a limited time) even without infrastructure, so are a good match for AllNet.
- Distributed network access. It must be possible to join and use the network without permission or approval from a central authority.
- Usefulness at low bit rates and high latencies. Sometimes ad-hoc and delay-tolerant networking is all that is available, an the network must be useful in these circumstances. When better performance is available through the infrastructure or direct connections, the network should be able to take advantage of the higher speeds and lower latencies.
- Security. Since ad-hoc technology uses untrusted intermediaries to deliver packets, all personal com-

munication must be encrypted end-to-end. Public communications can be sent in the clear.

- Authentication. Once I know my contacts' public keys, I can easily tell whether a signed packet is from one of them or not.

- Social Network. Pervasive authentication of known contacts makes it easy for my devices to keep track of my social networks.

The first two of these features suggests that, unlike the current Internet, addressing should not be based on the point of attachment to the network. In AllNet, addresses are self-selected random bitstrings. Delivery is by a combination of limited wireless broadcasts, sending to designated Internet hosts, and network nodes self-organizing into Distributed Hash Tables (DHTs).

While multiple devices might by chance (or maliciously) select the same random address, in AllNet such collisions just mean that the packet might be physically delivered to multiple destination devices. If a device receives a packet that it cannot decrypt, or that is not signed by a contact in the owner's social network, the device automatically treats this as any other packet not intended for this device.

The third feature suggests that one of the applications of AllNet should be a persistent chat system similar to SMS. SMS already provides delay tolerant low-bandwidth communications, but requires cellular infrastructure and is sometimes unavailable to individuals because of the way it is priced. With AllNet, such communication would take place over the Internet when available, and by ad-hoc and delay-tolerant networking when these are available.

For example, an AllNet chat message might be delivered in much less than a second if both devices are connected to the Internet. The recipient needs either a public IP address, or must request packet forwarding from another computer that has a public IP address. In either case, this public IP address must be known to the sender. If the sender does not know an IP address for this receiver, the sender forwards the packet to any node forming an Internet-wide DHT. If the recipient has requested packet delivery from DHT nodes corresponding to its address(es), the packet is delivered promptly.

If ad-hoc networking is available, the same packet will also be sent to the devices within reach of the sender. Each such device forwards the packet if that can be done within strict limitations on battery and bandwidth usage. If the destination can be reached through ad-hoc networking, the packet might be delivered in less than a second, or after a delay of many minutes if one or more of the intermediate and final devices turn off their radios part of the time.

Finally, the device will store the packet for a while, and make it available on request. To limit buffer size, packets are stored for a limited time, with newer packets generally replacing older packets, and packets for destinations known from the social network being stored longer than packets for strangers. If a packet is still present in the device when the packet's destination is in range, the packet is delivered at that time, whether that be a few seconds or a few days later.

To be useful with such delay variability, each chat message is tagged with a unique sequence number[1] and the time of transmission. The destination chat program uses the sequence number and timestamp to discard duplicate packets. Reusing a sequence number with a later timestamp allows the sender to request that a message be amended or deleted. The recipient is free to either honor or disregard such requests, but usually the chat program shows the latest version with an indication that older versions are available.

This chat protocol is to date the best developed of the applications of AllNet. Another significant application would allow devices that are not connected to the Internet to request web or email transfers through other AllNet devices that are connected and willing to share their access.

AllNet provides a way for two people who are making contact to securely exchange public keys. The basic mechanism is to transmit in the clear the public key, together with an HMAC of the key and a short secret string that the two parties have exchanged. Communicating the secret string is easy if the two can talk directly to each other or have in common a trusted friend. In the absence of these, the secret string can be communicated by other secure mechanisms such as a telephone call.

We have designed and developed AllNet beginning in the first half of 2012. Although the design is still evolving, two preliminary Linux versions (version 0 and version 1) have been completed, and the implementation of version 2 is underway. We expect that version 2 will be sufficiently useful to see initial use among the public at large, and we plan to port this implementation to a number of mobile platforms.

The next section is the main section of this paper, and describes in detail the design of AllNet. Section 3 describes the current performance and other interesting features of AllNet. Section 4 surveys related work, including a paper on AllNet previously presented at a conference, other projects that overlap with AllNet, and technologies that our design builds on. Section 5 reviews present status, future work, and gives concluding remarks.

[1]Sequence numbers need only be unique within any given conversation, so 48 bits are sufficient.

2. DESIGN

The design of AllNet includes a number of components. We begin by describing (Section 2.1) the packet forwarding algorithm, a novel and simple mechanism that combines existing approaches to make AllNet effective at delivering data under a variety of circumstances.

AllNet is specifically designed to keep resource usage below a specific, very low level for traffic that does not directly benefit the owner of the device or the owner's friends. Essential components of this design include a low power wireless forwarding algorithm (Section 2.2) and a packet prioritization scheme (Section 2.3). To distinguish friends and assign them a greater share of resources, AllNet provides an algorithm for keeping track of the social network of the owner of the device and anonymously sharing it with others (Section 2.4).

Finally, AllNet packets are designed to be public-key encrypted. Rather than using certificate authorities or a web of trust, the key exchange mechanism of AllNet (Section 2.5) allows the exchange of keys among individuals who know each other or who know someone in common.

2.1 Addresses and Message Forwarding

AllNet transmission combines wireless ad-hoc broadcasting with Internet transmission.

Every packet carries a destination ID. These destination IDs are bitstrings up to 8 bytes long, usually selected at random.

Should two people select the same ID, resulting in a collision, the only consequence is that they might attempt to decrypt each other's packets. The decryption will not succeed, but the attempted decryption will waste some energy.

Each time an ID is sent, it is sent with a one-byte length field that declares the number of bits of the ID that should be considered valid.

These destination IDs are used:

- As indices into the Distributed Hash Table. If I use a destination ID I, it is usually to my advantage to listen for packets from the DHT nodes that are responsible for I.
- To avoid attempting to decrypt packets that are clearly not for me.
- To prioritize decryption of packets that may be for me, giving higher priority to the ones that have more valid bits.
- To find out whether a packet is from an individual in my social network.

Each node generating or forwarding a packet sends it by:

1. broadcast on locally connected networks
2. broadcast to listeners
3. send to the DHT node(s) corresponding to the destination ID
4. if on the Internet and we have mappings of rendezvous points (RPs) for this destination ID, send to the RPs

In each case, the forwarding is subject to AllNet resource limitations. As long as these limitations are not reached, each packet is forwarded to all local nets, to all listeners, and to all DHT nodes and RPs corresponding to the destination ID. Otherwise, only higher-priority packets are forwarded, as described in Section 2.3.

A listener establishes a TCP connection to any other AllNet node. A node wanting to receive packets, on the Internet but not itself part of the DHT, and perhaps behind a firewall, might want to listen to several of the DHT nodes responsible for the parts of the address space corresponding to the node's own destination addresses.

Well-behaving nodes that are part of the DHT both receive the packets themselves (handing them to local applications), and forward the packets to each listener in accordance with step 2. In this way, a node that is not in the DHT may receive all its packets by being a listener to a DHT node.

The fourth way of forwarding packets deserves closer examination. It is only available when a device can send Internet packets directly. Before transmission, the device must have been given the IP and port number of a machine with a stable, publicly routable IP address that the receiver will connect to to retrieve its packets. The machine with that IP address is a Rendezvous Point or RP. A functioning RP will forward packets to the receiver, which may happen either by prior agreement, or by the receiver connecting as a listener to the RP.

A receiver may have multiple RPs that it listens to, and give a subset of these RPs to any potential sender. Having multiple RPs makes the system more fault-tolerant, and giving different subsets to different groups makes the network more resistant to traffic analysis.

A device may or may not have any mappings to RPs for a given destination address. If it does not, only the first three steps of forwarding are used. If it has many such mappings, perhaps because the destination address has few valid bits, the forwarding node sends to a randomly selected subset of these.

In general, the usage of stable RPs is preferred to using the DHT, and especially more so if the RP belongs to and is under the control of personal friends. RPs somewhat resemble mail servers, allowing communication between a sender and a receiver that may not have stable IP addresses or even be consistently connected to the Internet.

Symbol	meaning
p	target duty cycle for radio
p'	duty cycle for high priority
t	time to cycle the radio and announce
t'	time for packet transmission
B	transmission bitrate

Table 1: Symbols used in Section 2.2.

If an ack is sent in response to a received packet, the ack may carry, encrypted, the IP and port of the RP from which the packet was first received. This may be used by the sender when sending further packets to the same destination, prioritizing RPs deliver quickly.

Listening to nodes in the DHT corresponding to my address is useful as a backup when no RPs can be identified, and is essential when first connecting to AllNet. It is the way any node can pick an arbitrary string and automatically have a public routable address.

The design of the DHT is modeled after the DHT in Kademlia[1].

2.2 Sleep and Wake Cycles for Wireless

There are different kinds of traffic in AllNet. One of the important distinctions are between traffic that I (the owner of the device) have originated, traffic sent by my friends, and other traffic.

In the first case, there should be few if any resource limitations. The radio should be on whenever I wish to send packets, and also whenever it is likely that I will receive a response I am interested in.

For example, a cellphone walkie-talkie application could keep the radio on at all times, possibly discharging the battery relatively quickly to support low-latency and high-bandwidth communication among devices that are within range. But when my device is only forwarding packets on behalf of others, I am likely to want to limit the amount of battery energy used for forwarding.

One goal in the design of AllNet is that the network should be useful even if resource usage is limited to about 1% of the total available on each device. Recognizing that wireless transmission and reception may consume significant power, this means the radio should be off (or available for other uses) 99% of the time. More in general, we consider duty cycles (fraction of the time that the radio is on) of p or less, so the radio is off for fraction $1 - p$ of the time.

AllNet follows the general scheme of the Block Transfer Protocol [2] by synchronizing senders and receivers, then sending multiple packets one after another once the sender and receiver have synchronized.[2]

[2]A simpler, unsynchronized scheme, in which receivers listen fraction p of the time and senders send each packet $1/p$ times, is inefficient when p is small.

To synchronize, any device with packets to send (a sender) listens for announcements from other devices that wish to receive packets (receivers). Once a sender has heard from a receiver, a brief exchange similar to RTS/CTS provides some protection against collisions. The sender can then send a number of packets to this receiver. Other receivers within range may also receive the same transmission.

If a receiver takes time t to turn on its radio, transmit its announcement, wait for a reply, and if nothing heard, turn the radio off again, a receiver with duty cycle at most p must sleep at least time t/p between announcements. Accordingly, a sender wishing to reliably hear announcements should listen for at least time t/p.

A sender has more information than a receiver, and in particular, can tell whether packets in its queue have high priority. For such packets a sender may use a duty cycle $p' \geq p$, where perhaps $p' = 1$ for the sender's own packets, and $1 \geq p' \geq p$ for packets sent by the sender's friends.

If the sender listens for t/p and turns off the radio for time $(1 - p')t/p$ to give a duty cycle p', then the worst-case one-hop latency for a packet sent to a neighbor within range is $lat_{worst} = t/(p \times p')$.

If the bit rate of packet transmission is B and, once two systems are synchronized, bits are sent for time t', the overall rate of packet transmission is $B \times t'/(t/p)$, or pBt'/t. For example, if $p = 0.01 = 1\%$, $t = 0.1s$, $t' = 0.2s$, and $B = 10Mb/s$, the overall rate of transmission would be 200,000 bits/second, or 20 packets of 10,000 bits each.

If the sender has high priority packets to send and keeps its radio on all the time ($p' = 1$), it will overhear each potential receiver as soon as the receiver transmits, after $t/2p$ on average and t/p in the worst case. In the example above, the receiver will send an announcement on average after $5s$, and always within $10s$.

If the sender only has low priority packets to send ($p' = p = 0.01$), then the latency is $500s = 8m20s$ on average, and never more than twice as much, $16m40s$.

For senders that have experienced low rates of packets transmission or have very charged batteries, they may temporarily use a higher p' to reduce the transmission latency even when forwarding data for strangers.

The receiver can specify in its announcement the time t' it is willing to keep its radio on for receiving. If $t' > t$, the receiver may observe its overall duty cycle p by sleeping for t'/c (instead of t/c) after receiving packets for time t'.

To summarize, with this synchronization scheme senders may send about $O(p' \times p \times bandwidth)$ to any given receiver, and the average latency would be $t/(2 \times p \times p')$. Latency is minimized by minimizing t, that is, turning the radio on and off as quickly as possible.

2.3 Message Prioritization

When traffic is light, AllNet eventually forwards all packets it receives whose hop count has not expired.

When traffic is heavy, senders must decide whether and when to send packets. AllNet suggests that every device prioritize the packets it sends, with highest priority given to the packets the device's owner wishes like to send, and descending priority to packets for the owner's friends and then packets that benefit the network as a whole. This last is hard to determine, but AllNet suggests heuristics to favor some kinds of packets over others. None of these heuristics are required for participation in AllNet, but supporting them may improve the performance of the network as a whole.

2.3.1 Priority Computation

AllNet automatically prioritizes packets based on local information available to the forwarding node.

The priority is a real number between 0 and 1, internally represented in fixed point notation as an integer between 0 and 2^{30}.

To favor packets generated by the local system, local applications are allowed to specify their own priority for outgoing traffic. If not specified by the application, the priority of local packets defaults to 0.875.

Similarly, if a packet carries a sender ID and a matching certificate identifying one of the people to whom I have agreed to give resources (a friend), these packets are given a priority of 0.75. Certificates are discussed in Section 2.5.1.

For all other packets, a variety of independent priorities P_i are computed based on different information, then combined by multiplying them together. Since each $P_i \leq 1$, any factor that produces a low individual priority gives a low overall priority for the packet.

$$Priority = \prod_i P_i \tag{1}$$

2.3.2 Priority Factor from Social Distance

The social distance d is defined to be $d = 1$ for my friends, $d = 2$ for their friends, $d = 3$ for their friends, and so on. The social network used to keep track of social distances is described in Section 2.4.

When the social distance $d > 1$ of the sender is known, it is used to compute a social priority factor $P_s = 2^{1-d}$. As it should, this priority drops quickly with social distance. The size of a social network may be expected to grow nearly exponentially with social distance, so the computation of P_s exponentially decreases the priority with increases in social distance. Also, $P_s \leq 0.5$ for $d \geq 2$, so the priority of friends of friends is always less than the priority for friends.

If the social distance is not known, and if AllNet keeps track of the social network up to distance $d = n$, the distance used for someone who does not appear in the social network is $d = n + 1$. The current design keeps track of identifiers up to distance $d = 3$, so a stranger is arbitrarily assigned $d = 4$, giving $P_s = 2^{-3} = 0.125$.

Symbol	meaning
P_i	component of priority computation
d	social distance
P_s	priority from social distance
h	hops traveled
P_h	priority from hops traveled
g	hops to **g**o
P_g	priority from hops to go
r	rate of traffic from one sender
P_r	priority from traffic rate
b	valid bits in a packet address
P_b	priority from number of valid bits

Table 2: Symbols used in Section 2.3.

2.3.3 Other Priority Factors

The remaining P_i factors in equation (1) are based on:

P_m the maximum number m of times a packet may be forwarded. This field is set by the original sender and never changes. Since traffic with lower m is likely to require exponentially less network resources, $P_m = 2^{1-m}$.

P_h the number of hops h already traveled. There is some benefit to favoring packets that have traveled longer distances, namely that a retransmission would be more expensive than for packets that have not traveled as far. However, these packets have already had more chances to reach their destination, and in general it is wiser to favor local traffic, so we use $P_h = 1 - (h-1)/8$ for $h \leq 4$, and $P_h = 0.5$ for $h > 4$.

P_r the rate r at which packets from the same sender as this packet has recently been received. This factor discriminates in favor of known senders from which we have not forwarded many packets recently. Unknown senders have $P_r = 0.5$, whereas a known senders that has recently used fraction r of the bandwidth gets $P_r = 1 - r/2 \geq 0.5$.

P_b the number of bits b in the destination address. Again, there are a number of reasonable functions, and by default AllNet uses $P_b = 1 - 2^{-b}/2$. This function gives $P_b = 0.5$ for $b = 0$, $P_b = 0.75$ for $b = 1$, $P_b = 0.875$ for $b = 2$, and so on, reflecting the fraction of recipients who will not attempt to decrypt this packet.

These functions use only local information and information obtained from the packet being forwarded.

Even in the absence of social network information, these priority factors favor and encourage transmission of packets that will consume the least possible network resources.

2.3.4 *Priority Forwarding*

Since packets are sent in strict priority order, and since devices generally limit how many packets they will send, low-priority packets are more likely to be dropped than higher-priority packets. Packets forwarded over ad-hoc links are also likely to experience higher latency, as described in Section 2.2.

Most packet networks have idle periods. During such times all queued packets may be sent, including low-priority packets. Since low priority packets may have to wait for aless busy time, they may experience greater latencies than higher priority packets.

Lowering the priority of a packet therefore increases the expected latency and jitter (latency variation), as well as the expected packet loss rate. This should encourage application designers to send packets that are likely to be given as high priority as possible, for example by sending packets limited to as few hops as will reach the destination.

Closeness within the social network also affects priority. The next section explains the design of the distributed social network mechanism of AllNet.

2.4 Anonymous Social Network

Distributed social networks have been and continue to actively be developed, including for example Diaspora [3], Friendica [4], and DiSo [5] (Distributed Social Network). As for AllNet, the goal in these networks is to foster decentralized interpersonal communication. AllNet has the additional goal to continue communicating effectively even in the absence of infrastructure, and specifically, to allow devices that forward packets to determine the degree of connectedness within the social network and therefore the extent to which to devote limited resources to forwarding each packet.[3] To the extent possible, this should be done without revealing to others the identity of my friends.

The social network graph is easily built in a distributed fashion in a manner analogous with link-state routing. When I connect with a new person, for example Alice, I can send her a list of information about individuals in my social network, including for example Bob, Charlie, Donna and Eve, who are in the set f_{me} of my friends. This information should include destination addresses and public keys used to verify packets sent by the people in my social network. For example, it should include the address that I (and perhaps others) use when exchanging packets with Bob, and a corresponding public key that can be used to verify whether a packet really is from Bob. The information would not include Bob's name or other personally identifiable information.

[3]A more conventional distributed social network could be built as an application that uses the AllNet protocol.

Symbol	meaning
f_D	the set of friends of the owner of D
f_D^2	the friends of friends of D
f_D^3	the friends of friends of friends of D
$\overline{\lvert f \rvert}$	the expected size of a set f

Table 3: Symbols used in Section 2.4.

If Alice (or anyone she shares information with) already has contact information for Bob, she can tell that Bob and I are in each others' social network. If she does not know Bob, all she has are bit strings that she can associate with one of my friends, without knowing who that friend might be. In other words, she can add the set f_{me} into the set of her friends of friends, f_A^2. If I send her information about the set of my friends of friends, f_{me}^2, she can add the information to her set of friends of friends of friends, f_A^3.

When I forward to Alice information about f_{me}, I only include the public key and address. Alice can use these to verify that packets my friends' send are indeed from one of my friends (e.g. Bob), even without knowing who the friend is.

When I send Alice information about f_{me}^2, I send her only the initial few bits of each destination address, which she adds in her f_A^3. For example, this may include the first few bits of the destination address used by Bob's friend Frank, who is in f_{me}^2.

2.4.1 *Calculating Social Distances*

There may be times when a user might wish to prove to a stranger that their social connection is of distance at most d. For example, assume that Alice and George have never met, but Alice is looking for somebody to help her connect to the Internet, and George's device is the only one within range. We assume that Alice does not know which of the persons around her owns the device with which her own mobile device is communicating.

Further assume that Alice and George each have a distant connection to Helen, who is in Alice's f_A^p and in George's f_G^q, The social distance between Alice and George is then $p + q$. Alice might benefit by claiming that her connection to Helen is $p' < p$, but the information she has does not allow her to prove such a claim. Instead, she can prove to George that Helen is in f_A^p. George sends her a nonce N, and Alice returns the HMAC of information she has about Helen, using N as

a key. As long as $q \leq p$, George has all the information needed to verify that Helen indeed is in Alice's f_A^p.

If $q > p$, that is, the distance from Helen to George is greater than the distance from Helen to Alice, then there must be some other person (a friend of Helen's) for whom both Alice and George hold information, and who is closer to George than Helen is. In that case, Alice and George need to find this person so Alice can prove to George's satisfaction that this person is in her social network.

To search for people in both social networks which are no closer to Alice than to George, without divulging the details of the social network, Alice sends to George a Bloom filter reporting the distribution of the hashes of the information she holds about each individual. George finds any matches within his own social network, hashes them with the nonce, and sends them Alice part of each hashes. Since the Bloom filter is an imprecise data structure, not all of George's matches will be in Alice's social network, but when they are, she can prove it by sending to George the other half of the hash.

To see that this works in general, it is important to understand why people are generally closely connected, the anecdotal "Five Degrees of Separation". It is known that random graphs [6] have small diameters, that is, the maximum distance between any two nodes in the graph grows much more slowly than the number of nodes in the graph. In order for a social network to behave as a random graph, any two persons who are friends with each other must also each have other friends who are not friends in common.

To see how quickly the graph scales, assume that every person has about 100 random friends, so $\overline{|f|} \approx 100$. In reality, many people might have much more than 100 friends, but many of these will also be friends of friends rather than selected at random. In any case, with this assumption, the expected value of the number of friends of friends is $\overline{|f^2|} \approx 10^4$, and $\overline{|f^3|} \approx 10^6$.

Then, if I compare my f, f^2, and f^3 with a stranger's f, f^2, and f^3, we are likely to find at least one person in common as long as the number of people in the entire social network is less than $\overline{|f^3|} \times \overline{|f^3|} \approx 10^{12}$. This number is more than 100 times larger than the current and foreseeable human population, which is less than 10^{10}.

In short, relatively small social networks that maintain information for up to about a million people in f, f^2, and f^3 are often sufficient for any two people to determine the degree of their connectedness. This is directly related to the Birthday Paradox [7].

2.5 Secure Public Key Exchange

On the World Wide Web, secure exchange of public keys is mediated by a centralized Public Key Infrastructure (PKI) that depends on a number of trusted certificate authorities (CAs). This PKI is used routinely and is extremely reliable as long as the CAs can indeed be trusted, which unfortunately is not always true [8].

Symbol	meaning
s	a secret known by two people
E	the entropy of s, in bits
$HMAC_x(v)$	HMAC of v using x as key
PK_D	D's public key

Table 4: Symbols used in Section 2.5.

The Web of Trust, introduced by Zimmerman for PGP [9], allows individual users to certify other users. A recipient Ian of a public key alleged to be from Juliet may trust that this is indeed Juliet's key if the key is signed by Ken or Linda, whom Ian trusts to certify keys. While the decentralized nature of the Web of Trust makes it perfectly suitable for AllNet, this model is still somewhat more heavyweight than needed within a social network.

In a true social network, people generally know each other informally, and frequently have out-of-band ways of exchanging information.

For example, when Michael and I met at a social event, I was able to give him my contact information using only my voice, rather than a secure network. I have never checked Michael's ID, so I may trust him with my secrets, but not with my money. If I later learn that Michael is in fact Norm, I might still continue to trust him with my secrets. It is challenging to use the Web of Trust correctly because human trust is very nuanced, evolving, and implicit, which is hard and tedious for people to encode in computer software.

If Michael and I want to communicate using AllNet, when we meet at the party we exchange a shared secret s, randomly generated by one mobile device and entered into the other mobile device. If I enter the s displayed on Michael's device, my device can send my public key PK_{me} and an HMAC $H = HMAC_s(PK_{me})$. Michael's device, on receiving a key exchange packet (PK', H'), can verify whether $HMAC_s(PK') = H'$. If so, Michael can be confident that $PK' = PK_{me}$ is my public key. Only a sender familiar with s can generate a valid key exchange packet.

The HMAC computation may use any common hash function without substantially affecting this exchange.

The shared secret s is a nonce, usually randomly generated by one of the mobile devices, and useless to an attacker once the key exchange is complete. So s can be short and easily communicated as long as there is little risk of accepting a packet from an attacker. If s has about E bits of entropy, to avoid birthday paradox attacks, the packet carrying the legitimate new key should arrive well before $2^{E/2}$ other key exchange pack-

ets have been received (and if not, the exchange should be restarted with a higher value of E). There are several ways to accomplish this:

- When two parties are in direct transmission range, packets with hop count $h > 1$ can be discarded, allowing s to be very short.
- When two parties are not in direct contact, but do have an out-of-band mechanism for exchanging s, s should be longer and harder to guess.

Once the exchange has taken place in one direction, the response can be encrypted with the newly received public key, and explicitly carry the sender's own public key. This response can also carry a user profile, security information, and information about one's social network.

If there is no way to exchange out-of-band data, but I want to exchange keys with someone who is a friend of a friend, I can use a mechanism similar to the PGP Web of Trust and have the intermediate friend securely forward our public keys to each other.

2.5.1 Certifying Packets

.

Section 2.3 discussed ways to prioritize packets based in part on the social distance between the owner of the forwarding device and the owner of the sender address. Section 2.4 described how to track the social distance of sender addresses beyond the circle of friends. If packets are to be prioritized based on this social distance, a forwarding device must be able to inexpensively verify that the packet was indeed sent by the claimed sender. To do this, the sender must place into the header a 256-bit digital signature of the encrypted packet body, signed with the sender's private key. The corresponding public key is distributed through the social network described in Section 2.4.

Successful certification proves that the sender address in the packet header corresponds to the (possibly encrypted) packet contents. For an attacker to forge the sender address, the attacker must be able to produce valid signatures corresponding to a given public key. The current version of AllNet includes in the packet header a 256-bit digital signature. This is designed for an ECDSA `secp128r1` signature, but other signatures that fit in 256 bits can also be used.

Packet certification is not always necessary, and uncertified packets are well supported by AllNet, although sent with lower priority. If a sender does not appear with the social network of the forwarding device, the packet will also be sent with lower priority.

Like sending a packet, verifying a signature consumes a certain amount of CPU time and energy. In our tests, verifying a 128-bit ECDSA signature took 0.3ms on a modern processor, and less than 3ms even on an 800-MHz celeron. Since a high priority packet may be forwarded many times, the energy to verify the signature can be less than the energy to forward the packet.

3. PERFORMANCE AND EVALUATION

3.1 Latency

Since AllNet is designed primarily for traffic such as chat that normally requires low bit rates, latency is more important than throughput. Latency varies dramatically depending on the circumstances of the communicating hosts. Each of the next sections considers a different scenario.

3.1.1 Applications on the same system

To obtain a baseline measurement, we wrote a simple AllNet client to send a 200-byte packet via the AllNet daemon, and record the time (using `gettimeofday`) until the response was returned. The AllNet daemon returns all packets to all local listeners, and therefore all applications can expect to get back every packet they send.

On one system (system A), a 2.5GHz dual-core pentium E5200 running 64-bit Ubuntu Linux 12.04, over 10 trials, the time to receive the message varied from a minimum of 0.250ms to a maximum of 0.380ms, with a mean of 0.289ms and a median of 0.273ms.

The same setup tested on a commercial virtual machine (system B) had times ranging from 0.443ms to 0.754ms, with a mean and median of 0.580ms.

3.1.2 Direct internet connection

The same simple client was modified to add a mapping (as described in Section 2.1) to an RP across the Internet, so that packets would also be forwarded to one other system across the Internet. We also modified the client so the recipient would immediately reply. The two systems in this test were the same as in the previous test.

With this setup, the times ranged from 124ms to 128ms with both mean and median of 127ms from system A to system B, and from 125ms to 128ms with both mean and median of 126ms from system B to system A.

For comparison, ping sent 10 packets (56-byte payloads) between the two systems. From system A to system B, ping reported an average round-trip time of 84.873ms, and in the opposite direction 84.846ms, in both cases with variation (`mdev`) of less than 0.5ms.

A discussion of this performance is in Section 3.1.4.

3.1.3 Internet connection via another node

To further test forwarding over the Internet, we added a third device (system C), a 930MHz Pentium-III running 32-bit Ubuntu Linux 12.04, which is on the same

network as system A (ping times from system A average 0.189ms). The two daemons on systems A and B were given a mapping to systems C for the destination address of the packets.

With this arrangement, the times were very similar to the times with the direct connection, ranging from 126ms to 132ms (both mean and median of 129ms) from system A to system B, and from 124ms to 129ms (mean of 127ms and median of 128ms) from system B to system A.

We verified that packets were indeed being sent over system C by stopping the daemon on system C and finding that packets no longer made it to the other side.

3.1.4 Pure ad-hoc forwarding

We also tested sending packets over wireless ad-hoc networks. Again, this implementation has yet to be optimized. For example, the interface is placed into ad-hoc mode by calling `iw` and `ifconfig`. In 10 tests on system A, calling `iw` and `ifconfig` to turn on the interface and place it in ad-hoc mode, then turning the interface off again, took between 98ms and 190ms, with a mean of 160ms and a median of 164ms.

Because of this substantial time to turn the interface on and off, we tested the latency of the wireless system without ever turning off the wireless interface. This test resembles the previous tests, but the packets were sent from system A and returned by system D, a 1.2GHz Core 2 Duo running 64-bit Ubuntu Linux 12.04.

In this test, packets were sent wirelessly over a direct connection using 802.11 ad-hoc mode.

The round-trip times ranged from 44ms to 60ms with mean of 48ms and median of 47ms from system A to system D, and from 17ms to 58ms with mean of 41ms and median of 45ms from system D to system A.

On the same connection, `ping` reported round-trip times between 2.4ms and 15.2ms, averaging to 6.2ms with `mdev` of 4.3ms.

For this test, as for the test over the Internet connections, round trips took about 42ms more than `ping` round trips, though in this case AllNet had a two round trips that took less than 20ms. There must be something in the software that causes this delay, but not for every packet. It remains for future work to find and remedy this delay.

We also tested a mixed network, with system D sending packets wirelessly to system A, which forwarded the packets over a low-latency internet connection to system C. The times measured were similar to the times over the wireless connection alone.

3.2 Storage for Delay Tolerant Network

In a DTN, packets are delivered when a device carrying the packet is within range of the packet destination. The latency thus varies depending on the human choices that cause mobile devices to come within range, and can vary from minutes to days or longer. Ignoring the latency, we instead consider how much storage is needed to deliver packets.

Suppose two people and their devices meet occasionally. Once the devices are in range they immediately exchange all stored packets intended for each other and all packets intended for common acquaintances.

If every person in the DTN has F friends and sends mb bytes to each of these friends in between meetings, each device must have $mb \times F$ bytes of storage just for the packets sent by the owner of the device. If $F = 100$ and $mb = 100,000$ bytes (scenario 1), the device needs 10MB of storage for this data. If (scenario 2) the data contains multimedia so that $mb = 100$MB, the device needs 10GB for the owner's data.

To also store the messages for the owner's friends requires storing $mb \times F$ bytes for each of the F friends of the owner, or $mb \times F^2$. In scenario 1 this requires at least 1GB of storage, and in scenario 2 1TB.

Scenario 1 (textual data) seems very feasible today. Multimedia data (scenario 2) is feasible today if a device only stores its own data, and in the foreseeable future it may very well be possible to store even friends' multimedia.

3.3 Efficiency

Broadcast is typically seen as an inefficient technology because it can deliver packets to recipients that have no use for them.

Classical Ethernet [10], for example, unless the interface is in promiscuous mode, uses a hardware mechanism to efficiently discard received packets that do not match the receiver's address. Modern Ethernet has largely stopped broadcasting, unless the broadcasts is required as for ARP or DHCP.

Broadcast also has a few advantages. Addressing is optional on a broadcast network, as long as the recipient can figure out which packets to use. On a physical broadcast medium, multiple devices may obtain the same packet from a single transmission, although the benefits of this are only infrequently realized. A security benefit of broadcast is that a packet is delivered to its destination without revealing which of the receiving nodes is the destination.

Ad-Hoc networks have additional efficiency challenges. For example, a node receiving, then rebroadcasting a packet keeps the medium busy for at least two packet transmission times. Multiple nodes retransmitting the same message have to delay the transmission by a random amount to try to avoid collisions.

While future versions of AllNet may attempt to address some of these inefficiencies, already AllNet offers senders a choice between packets that are more likely to be delivered with high priority and low latency, and

packets that contain less information to thwart traffic analysis.

Further, AllNet puts a limit on how many resources will be used to forward packets. Even if efficiency is low, at worst this limits the number of AllNet packets that are successfully sent, rather than affect the resources (battery and spectrum) used for AllNet.

If AllNet is used mostly for interpersonal communication of text messages, the inefficiency is not likely to be a concern. It is only if larger amounts of data are exchanged over the wireless medium that the design of AllNet might need to be revised to improve the efficiency. Future work may explore the improvements in efficiency obtained by use of routing protocols and of scheduled wireless transmissions.

3.4 Security Considerations

All user data in AllNet that is not sent in the clear, is encrypted and signed, so the data is kept confidential and the recipient knows that the sender has a private key corresponding to the public key of one of its contacts. Unless the keys are compromised (or the algorithms are broken), the data is secure. AllNet applications currently use 4,096-bit RSA keys, but this choice can be changed easily and without impacting the underlying implementation of AllNet.

Good key management practices require that keys be backed up securely. For AllNet, we plan to allow one device to advertise as its own multiple keys, including both multiple keys on the same device, and also other keys on other devices belonging to the same owner. Anyone sending to this owner would normally send to all the owner's devices. Recipients of packets signed by any of the keys can trust that the packet is sent by the owner of all the keys. In case one device is compromised, the other device(s) can still send secure and authenticated packets to the owner's contacts, informing them of the situation and helping to alleviate any problems.

Data that is sent in the clear is not kept confidential. While further experience is needed, we expect that data sent in the clear will normally be ignored by users, unless (a) it is signed by a governmental authority using a recognizable key, or (b) the user is searching for a business nearby, has recognized an emergency situation, or has other reasons for wanting to read packets from strangers. In any case, the software can clearly mark for the user which packets are and are not authenticated.

Assuming that these mechanisms function as designed, remaining security challenges include traffic analysis and denial of service attacks. The wireless medium is particularly vulnerable to these attacks, since attackers can overhear or inject traffic without a physical connection.

As mentioned in Section 3.3, in AllNet it is up to the sender of a packet to choose between packets that are more secure and packets that are more likely to be delivered. More secure packets give fewer or no bits of the sender or destination ID, and have no certificate to indicate who might have sent them. Both of these features make it harder for a listener to figure out where the packet is coming from, and therefore help resist traffic analysis. There may be circumstances where this is appropriate, and other circumstances where it is preferable to be vulnerable to traffic analysis but have a higher probability that the packet will reach its destination.

Interestingly, denial of service is in some ways the exact opposite from traffic analysis. The denial of service attack can usually be stopped if the source of the attack is found, so an attacker is likely to want to send untraceable packets as much as possible. However, AllNet forwards these very packets only after all higher priority traffic has been sent, giving the attacker a choice between traceability and ineffectiveness. As a result, an effective denial of service attack probably requires, as it often does at present, taking over devices belonging to others. With AllNet, even this is not a very good strategy for the attacker, since the recipient of the attack can use the authentication to identify the sending device.

One final security consideration is the concern that if mobile devices are used as identification and keys to access resources, they become more valuable targets of attack. Fortunately, manufacturers of mobile devices and mobile operating systems seem to be familiar with the issue of securing devices, and continued progress in this area may be expected.

3.5 Ethical Considerations

Any powerful technology, including encryption, can be used for positive as well as negative purposes. Encryption is widely used by very diverse people, including bankers, criminals, whistleblowers, terrorists, military, and human rights campaigners.

To the extent that AllNet is successful and properly implemented and used, it will provide a large number of individuals with the ability to hold private conversations with any number of individuals they have made contact with. It will give recipients the technical ability to discriminate based on the sender of packets, if known, or based on the sender not being known. The only way for even legitimate governments to directly obtain this information would be to run software on the mobile device or obtain the physical device and defeat its security.

The decentralization of network function should make it much more difficult than at present for whoever controls the infrastructure to prevent others from communicating with their peers or to control the contents of the communication.

There are many ethical considerations to transferring to individual control the power over one's own commu-

nications. Without trying to provide an answer to the age-old tension that more or less balances centralized and individual powers, community and freedom, it is worth pointing out some of the advantages of personal control over personal communications:

- Privacy is enhanced when fewer people can control or intercept others' communications.
- Freeing the means of communication means people may communicate without as much concern for economic or political considerations.
- It is harder for a powerful person or government to spy on or interrupt the communications of weaker persons.
- Decentralized selection of IDs make it harder to automatically associate packets with people.

There are also related disadvantages:

- Even legitimate governments (or parents) may be unable to reveal information that would lead to social (or family) benefits.
- The relative anonymity of AllNet might in some cases, and especially in the case of unencrypted messages, lead to a lack of accountability that could favor abusive use of the network, including cyberbullying. Such abusive use is less of an issue if unauthenticated messages are ignored.
- When non-experts maintain a secure system, the chances of accidental loss of security or loss of data are higher than when experts provide the security.

4. RELATED WORK

Most of the related work used in the design of AllNet was described in Section 2. This section compares AllNet to similar projects, for each briefly outlining similarities and then focusing on substantial differences. Section 4.5 describes a previous publication on AllNet, and outlines the substantial changes made to the project since that time and the substantial differences between this paper and the prior.

4.1 Ad-Hoc Networks, DTNs, and DHTs

There has been much research on Ad-Hoc Networks, beginning with the early work in MANETs [11] and including fundamental work on Ad-Hoc networks [12]. The design of AllNet builds on these and many more results.

There is a plethora of routing protocols for wireless networks. While this version of AllNet uses broadcasting instead of routing on the ad-hoc portion of the network, future updates may use protocols such as OLSR [13] or AODV [14].

Delay Tolerant Networks [15] have also been the subject of much research, which again the design of AllNet builds on.

The same is true for Distributed Hash Tables [16, 17, 18, 19]. The field of DHTs has had many developments to support active peer-to-peer communities, and many systems used on a daily basis. Although the DHT for AllNet is not yet implemented, we plan to follow the design of Kademlia [1] because of its flexibility and redundancy.

While AllNet builds on previous work in all these areas, it is unique in combining these areas to create a new network with the specific purpose of providing interpersonal communication both with and without the infrastructure.

4.2 Emergency Wireless Networking

Many people have known for a long time that wireless communications can be useful when the infrastructure fails, and especially in emergencies. Even before the era of digital wireless networks, amateur radio operators provided communications in case of earthquakes or other major disasters. The characteristics of these (often ad-hoc) systems shared by the wireless portion of AllNet include communication over low bit rate channels.

Specific recent projects in this field include Lifenet [20] and the CDAC TERA network [21, 22]. The latter is infrastructure-based and relatively expensive, but has the advantage of working with unmodified mobile devices and being available to relief agencies.

Lifenet, like AllNet, is designed to work well even without the infrastructure yet can use the infrastructure when available. Lifenet also has focused more than AllNet on good routing protocols to support larger ad-hoc networks and higher-bandwidth applications than would work well using only broadcast. In the future, we may adapt AllNet to use this routing protocol, called simply Flexible Routing.

Unlike Lifenet, AllNet focuses on providing at least low-bandwidth communication whenever possible. We also strongly believe that a protocol, to be useful in emergencies, should also be useful in daily lives. Only by having users accustomed to the software and prepared to use it under normal circumstances will they be able to make good use of it in an emergency.

To support daily use, the design of AllNet has focused strongly on security and the maintenance of social networks. Both security and social networks can be beneficial in emergencies as in daily lives. The development of specific applications is also essential to daily use.

4.3 Secure Decentralized Networks

Three main efforts towards providing secure, anonymous networks are Tor, Freenet, and Bitcoin.

Tor [23] is designed to provide anonymity and security for Internet access, and does so through onion routing, where messages are repeatedly encrypted and slowly decrypted as they make their way through the network. Routers may issue fake messages to try to defeat traffic analysis.

Unlike Tor, AllNet does not decrypt packets as they are forwarded, instead relying on end-to-end encryption between trusted hosts. Fundamentally, in Tor a user must, to a however minimal extent, trust the routers in the network, whereas in AllNet the user must trust his or her device and the device(s) of the recipient of the message. Trusting the routers might be a good strategy in a fixed network with known routers, but is not likely to work well in a dynamically changing ad-hoc network such as envisioned for AllNet.

Freenet [24] is a content distribution network designed to automatically distribute content while concealing both the source and consumers of the content. These anonymity goals resemble the anonymity goals of AllNet, where a device forwarding a packet may not know where the packet is coming from or who the intended recipient might be. Somewhat similar to the social network in AllNet, Freenet has a Darknet mode where communication is routed through people to whom one has manually set up a connection. Also like AllNet, Freenet uses a system similar to DHTs to locate data, and has a notion of broadcasting requests until they either arrive or time out.

Like Tor, and unlike AllNet, Freenet is designed primarily to be supported by infrastructure networks. Also, Freenet focuses more on storing and delivering immutable data objects rather than facilitating dynamic and time sensitive communication among individuals. Also, Freenet tries to optimize paths to specific destinations by bringing nodes closer together when they exchange information. In AllNet, nodes aware of how they can be reached explicitly communicate that information to other nodes in their social networks.

Intriguingly, the Bitcoin system [25] provides anonymity without resorting to encryption, and provides authentication without any need for personal identifiers.[4] Like AllNet, Freenet, and Tor, Bitcoin is completely decentralized. Like AllNet, data in Bitcoin is time sensitive but some delay is normal. Like all these other technologies, Bitcoin is most effective on systems that are well connected to the global Internet, but unlike these technologies, AllNet is designed to also work well with intermittent or no connection to the Internet.

And while the Bitcoin network can be used to store and communicate arbitrary information, its main purpose is to store and transfer value, and in that, Bitcoin is quite different from AllNet (and Tor and Freenet). The lack of identities and of a social network are further distinctions between Bitcoin and AllNet.

[4]Just as intriguingly, the original author of Bitcoin has managed to maintain his personal anonymity, being known only by the pseudonym Satoshi Nakamoto.

4.4 Interpersonal Communication Systems

A number of infrastructure-based systems provide human-to-human text based communication using short messages. The two most famous are Short Message Service, also known as SMS or text messaging, and Twitter. Both have been tremendously successful while severely limiting the length of individual messages.

These systems have provided substantial inspiration in the development of AllNet, forever reminding us that even short communications can be extremely useful.

Unlike AllNet, these services require infrastructure, are proprietary, and in the case of SMS, are often quite expensive. For example, a single SMS message requires fewer bits and lower quality of service than one second of voice calling, but is sometimes priced higher than one minute of conversation.

And this is the other inspiration for AllNet: an awareness that such services should be available as widely as possible, without the expense and control that sometimes accompanies the need for infrastructure.

4.5 Previous Publication on AllNet

AllNet was first described at a conference in 2012 [26]. The substantial documentation and the source code (available under a BSD-style licence) have also been posted to the AllNet main web site [27].

The fundamental idea has not changed since then. We are developing a technology to support interpersonal communication over both the Internet and ad-hoc networks of personal mobile devices. However, we have done a lot of work to improve AllNet since that publication. In particular, the earlier paper described version 0 of the protocol, whereas this paper describes version 2. The changes are numerous, and though many are small, some are significant.

Version 2, for example, has a completely redesigned header and address. In version 0 there was no DHT and forwarding was by limited broadcast or to a specific address. The applications had to do source routing, forwarding by the AllNet daemon was very complicated, and addresses were complicated as well. Specific kinds of addresses in Version 0 included IPv4 and IPv6 addresses and several different addresses for different kind of broadcasts. Version 0 also required senders to know a lot about the topology of the network and how to reach a peer.

Replacing all these addresses with arbitrary bitstrings, and optional mappings from bitstrings to IP addresses, makes the system more robust and allows for improved performance when better information is available, for example about RPs. Likewise, the DHT improves per-

formance by reducing the need for broadcasting.

The structure of the AllNet daemon implementation is new and considerably simpler in Version 2, and the applications have been substantially redesigned as well. The key exchange mechanism is similar but has been improved by using HMAC instead of simple hashes, and the chat protocol has been improved by adding per-conversation persistent counters and timestamps to messages.

This paper also includes over 6 months' worth of experience with both implementing Version 1 and beginning to implement Version 2, as well testing and learning what works and what doesn't. None of this was available at the time the previous paper was written or presented. And because this paper is more substantial than the one presented in the conference (14 rather than 6 pages), we have been able to include a lot more information about how AllNet accomplishes its goals.

5. FUTURE WORK AND CONCLUSION

5.1 Implementation Status

This paper reflects the design and preliminary implementation of Version 2 of AllNet.

What has been implemented and tested is the AllNet daemon, which forwards packets based on priority both on the Internet and across wireless interfaces. This preliminary implementation includes dropping duplicate packets, supporting listeners and mappings, and forwarding cached packets on demand.

Functionality that has not yet been integrated into the Version 2 daemon includes the DHT, the social network, verifying packet signatures, and keeping wireless interfaces off most of the time.

In contrast, Version 1, which is not interoperable with Version 2, has a chat client and key exchange program. The chat client keeps data persistently and offers a very simple textual interface.

5.2 Future Applications of AllNet

While chat is a wonderful application to begin with, once AllNet is working well it might be useful for a number of similar purposes, including web access, email access, and generalized authentication.

The world wide web can be accessed using text-only browsers. This kind of access eliminates the need for high-speed connectivity. From personal experience, text-based Internet access, while much less stimulating than full multimedia access, is significantly better than no access at all. Many large Internet web sites, including google and wikipedia, provide good support for text-only access, and if more people were to access the Internet with text only, more web sites might provide better support. For AllNet to support web access, the crucial step is finding a device G that has access to the Internet and is willing to share some of this access.

When using secure http, intermediate devices such as G would be able to tell which websites were visited, but not the content of the transmissions.

Similar to web access is the issue of email access. When the Internet is not otherwise available, it can be very useful to get even basic email information such as one summary line per email received. When retrieving a message, a specific protocol can ensure that no attachments and only the first few bytes of the message are sent, unless the connection is fast or the data is explicitly requested.

Once a user's device has keys, and the user has become accustomed to managing them wisely, such keys could be used more widely for authentication, particularly for anything which currently requires a password. There are many applications where the basic data to be transferred need not be bulky, including online banking, remote login, and the submission of papers to conferences.

5.3 Future Work

The immediate goal of the AllNet development is to complete a chat application and a key exchange application essentially equivalent to what is available in Version 1. In addition, we plan to provide the other components needed for full functionality, including the DHT, the social network, verifying packet signatures, and turning off wireless interfaces most of the time.

After this, to make AllNet more attractive to users, we plan to improve the user interface of the chat application and port it to a variety of mobile platforms, especially Android and iOS, and desktop platforms besides Linux. We believe users will recognize the advantage of AllNet over other messaging applications that only support one or a few kinds of devices.

After we have completed a fully functional chat application, we can begin to develop and integrate new protocols and applications, particularly the web and email access, and more specialized protocols, for example to communicate to my social network my status, and especially the addresses of trusted Rendezvous Points.

There is also much work to be done to evaluate the protocol and implementation and improve them where possible. In particular, the social network makes it easy to offer interpersonal incentives to help others. It would be interesting to study how much people participate in AllNet to build a community and its complement of how much people participate for their own benefit, and perhaps, how much the two overlap.

5.4 Conclusion

It seems worthwhile to support exchanges of text messages among people whenever that can be done, with or without the infrastructure. Such low bit rate commu-

nication can be extremely useful in a number of cases, most dramatically in emergencies, but will only be widely adopted if it is useful for daily communications.

At this point, the design of AllNet is elegant and very effective, even though a lot remains to be done.

People often get excited about AllNet, believing that it will lead to a revolution in how everyone communicates. But even if AllNet is wildly successful, ad-hoc networks don't scale well. We will still need, use, pay for, and get the benefits of the infrastructure. We might see evolution in the market for Internet access rather than revolution.

What AllNet can and will do is set a baseline of providing interpersonal communication securely and for free whenever and wherever possible, motivated not by profit maximization but by the desire to help people communicate and to build better communities.

6. REFERENCES

[1] P. Maymounov and D. Mazières, "Kademlia: A Peer-to-peer Information System based on the XOR Metric", 1st Int'l Workshop on Peer-to-Peer Systems (IPTPS '02), March 2002, Cambridge, MA, USA.

[2] M. T. Hansen and E. Biagioni, "BTP: a Block Transfer Protocol for Delay Tolerant Wireless Sensor Networks", 5th IEEE Int'l Workshop on Practical Issues in Building Sensor Network Applications (SenseApp 2010), October 2012, Denver, CO.

[3] "Diasporia – All about the Diaspora Social Network", http://diasporial.com/whats-diaspora

[4] "friendica", http://friendica.com/

[5] "DiSo Project", http://diso-project.org/

[6] F. Chung and L. Lu, "The Average Distance in a Random Graph with Given Expected Degrees", Internet Mathematics, vol. 1 (2003).

[7] D. Bloom, "A Birthday Problem", American Mathematical Monthly, vol. 80 (1973).

[8] "Hackers issue fake security certificates for CIA, Google", Electronista, 05 September 2011, http://www.electronista.com/articles/11/09/05/diginotar.hack.tied.to.iranian.government/

[9] Philip Zimmermann, "Why I Wrote PGP", in the Original 1991 PGP User's Guide (updated in 1999).

[10] R. Metcalfe and D. Boggs, "Ethernet: distributed packet switching for local computer networks", CACM, vol. 19, July 1976.

[11] Broch, Maltz, Johnson, Hu, and Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols, Proceedings (ACM MOBICOM 98), October 1998.

[12] Gupta and Kumar, "The Capacity of Wireless Networks", IEEE Transactions on Information Theory, vol. 46, March 2000.

[13] Clausen and Jacquet, Editors, "Optimized Link State Routing Protocol (OLSR)", Experimental RFC 3626, Oct. 2003.

[14] Perkins, Belding-Royer, and Das, "Ad hoc On-Demand Distance Vector (AODV) Routing" Experimental RFC 3561, July 2003.

[15] Kevin Fall, "A Delay-Tolerant Network Architecture for Challenged Internets", SigCOMM, Aug 2003.

[16] Ratnasamy, Francis, Handley, Karp and Shenker, "A scalable content-addressable network", ACM SigCOMM 2001.

[17] Stoica, Morris, Karger, Kaashoek, and Balakrishnan, "Chord: A scalable peer-to-peer lookup service for Internet applications", ACM SigCOMM 2001.

[18] Rowstron and Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems", 18th Int'l Conf. on Distributed Systems Platforms, 2001.

[19] Zhao, Huang, Stribling, Rhea, Joseph, Kubiatowicz, "Tapestry: A Resilient Global-Scale Overlay for Service Deployment", IEEE JSAC, vol. 22, 2004.

[20] "About LifeNet", http://www.thelifenetwork.org/about.html

[21] Int'l Federation of Red Cross and Red Crescent Societies (IFRC), "TERA (Trilogy Emergency Relief Application) and Beneficiary Communication", http://www.ifrc.org /en /what-we-do /beneficiary-communications /tera/

[22] CDAC Network, "The CDAC Network: Improving the Effectiveness of Humanitarian Response", white paper, http://www.cdacnetwork.org

[23] R. Dingledine, N. Mathewson, P. Syverson, "Tor: The Second-Generation Onion Router", Usenix Security 2004.

[24] Ian Clarke, "A Distributed Decentralized Information Storage and Retrieval System", University of Edinburgh, 1999. https://freenetproject.org/papers/ddisrs.pdf

[25] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", made public May 24 2009. Available from http://bitcoin.org/bitcoin.pdf

[26] Edoardo Biagioni, "A Ubiquitous, Infrastructure-Free Network for Interpersonal Communication", 4th Int'l Conf. on Ubiquitous and Future Networks (ICUFN), July 2012.

[27] http://alnt.org/