Non-Hierarchical Networks for Censorship-Resistant Personal Communication

by

David Robinson Bild

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2014

Doctoral Committee:

Associate Professor Robert P. Dick, Chair
Associate Professor Jason Flinn
Associate Professor Z. Morley Mao
Professor Paul Resnick

2014

ACKNOWLEDGMENTS

I would like to thank my adviser, Professor Robert P. Dick, for his advice over the duration of my time as a graduate student. He planted the seeds from which this disseration grew. Thanks to his broad research interests, I have had the opportunity to work on a variety of interesting projects.

The work in this disseration was highly collaborative. Many thanks to Yue Liu for our numerous (and lengthy) discussions. She developed an early version of the Mason test and was instrumental in the design and implementation of Whisper, Manes, and Shout. Numerous undergraduates helped with implementation as well. Special thanks to David Adrian and Gulshan Singh for their work over several years. Thanks also to Nate Jones, Rongrong Tao, Jonathon Tiao, Anthony Tesija, and Junzhe Zhang for their hard work. And of course, thanks to the project advisers, Professor Robert P. Dick, Professor Z. Morley Mao, and Professor Dan S. Wallach, for providing guidance, making suggestions, and editing many, many paper drafts.

Thanks to Professor Jason Flinn and Professor Paul Resnick for serving on my committee. Your suggestions greatly improved several aspects of this work.

Thanks to everyone in our research group—Lan Bai, Xi Chen, Xuejing He, Phil Knag, Yue Liu, Yun Xiang, and Lide Zhang —not just for your professional collaboration, but your friendship as well.

Finally, I must thank my family for their continued and unwavering support. My parents have always encouraged my pursuits and I would not have completed this journey without them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The Internet promises widespread access to the world's collective information and fast communication among people, but common government censorship and spying undermines this potential. This censorship is facilitated by the Internet's hierarchical structure. Most traffic flows through routers owned by a small number of ISPs, who can be secretly coerced into aiding such efforts. Traditional crypographic defenses are confusing to common users. This thesis advocates direct removal of the underlying heirarchical infrastructure instead, replacing it with non-hierarchical networks. These networks lack such chokepoints, instead requiring would-be censors to control a substantial fraction of the participating devices—an expensive proposition.

We take four steps towards the development of practical non-hierarchical networks. (1) We first describe Whisper, a non-hierarchical mobile ad hoc network (MANET) architecture for personal communication among friends and family that resists censorship and surveillance. At its core are two novel techniques, an efficient routing scheme based on the predictability of human locations and a variant of onion-routing suitable for decentralized MANETs. (2) We describe the design and implementation of Shout, a MANET architecture for censorship-resistant, Twitter-like public microblogging. (3) We describe the Mason test, a method used to detect Sybil attacks in ad hoc networks in which trusted authorities are not available. (4) We characterize and model the aggregate behavior of Twitter users to enable simulation-based study of systems like Shout. We use our characterization of the retweet graph to analyze a novel spammer detection technique for Shout.

CHAPTER 1

Introduction

The Internet promises easy access to the world's information and communication among people. Discovery that used to require hours browsing shelves in a library or days watching a mailbox for materials from another institution now entails a simple web search, often completed in several minutes. Conversions with distant family members or colleagues endured the high latency and cost of physical letters and documents[1]. Wide-spread distribution of a new idea required significant capital backing, a distribution channel like a newspaper, radio station, or television show. For the niche, controversial, or unpopular, such distribution was often not available. The Internet, in principle, removes these limitations. Hosting and finding content on the World Wide Web is cheap enough that anyone who so desires[2] can participate. Email, instant messaging, and voice chat are ubiquitous and essentially free of cost.

This promise is only realized, though, when communication within the Internet is open. Intentional blocking of some traffic, e.g., identified by content, source, or destination, can hide specific knowledge from a significant fraction of entire populations and discourage those affected from engaging in thoughts or discussions deemed undesirable by the censors. Unfortunately[3], such censorship is commonplace and widespread, instituted by governments

[1]The telephone and fax machine also addressed this type of communication. All the problems of the Internet discussed in this thesis apply to the telephone network as well.

[2]This is not yet the case in severely economically-disadvantage areas, but the principle remains true. Much ongoing work is dedicated to bringing access to these places.

[3]We take as an axiom that both unfettered access to willingly-shared information and the ability to privately communicate with people of one's choosing is fundamentally a good thing. This view is widely debated, but

and network providers alike.

The Chinese government is likely the most familiar Internet censor to American audiences. In addition to blocking some content considered harmful to individuals, e.g., pornography, gambling, or violence, they censor information that might influence public opinion against the government and its actions, e.g., information about Tiananem Square protests, Taiwanese independence, and the Fulang Gong discipline [9]. Access to foreign news media that might report on such events like the BBC or Yahoo News is frequently blocked. Domestic social media channels like Weibo are selectively filtered [10] and foreign sites not implementing such filters like Facebook and Twitter are frequently blocked [11].

China is not alone. Classifications from the OpenNet Initiative [12] and Reporters Without Borders [13] label Iran, Syria, and North Korea, among others, as employing pervasive censorship. Other countries like Tunisia and Egypt experienced significant filtering before or during the 2011 Arab Spring uprisings [14–16]. As an extreme example, Egypt completely disabled Internet access for several days in February 2011 [17, 18].

Although providing historically unprecedented access to information and communication, the Internet also presents an unprecedented opportunity to surveil individuals for possible reprisal. Internet traffic monitoring is used to determine the interests, social relationships, and daily habits of individual users. Some of this analysis can be good[4], e.g., for recommending interesting content or better targeting advertisements, but much is also negative.

Examples abound. In December 2008, at least 56 online journalists were imprisoned on charges stemming from reporting on or voicing disagreement with government policies [19]. In China, citizens posting online essays criticizing government policies and exposing corruption were censored and sentenced to jail terms [20, 21]. In the United States, recent reports about so called *warrantless wiretapping* suggest that much telephone and Internet

both supporting and dissenting arguments are left to philosophers.

[4]It is still important to remember that even when the initial intent is good, the collected data might fall into more nefarious hands at a later date.

traffic is streamed through government facilities for monitoring [22]. Even worse, large data centers to store years worth of collected data are under construction [23, 24].

In its early years, people thought that the Internet's resilient routing protocols would resist censorship.

> The Net interprets censorship as damage and routes around it.
>
> (John Gilmore, 1993)

Unfortunately, this is untrue, as is now recognized.

> I used to believe the Internet offered limitless opportunities for free speech; now I believe it is becoming a smorgasbord of opportunities for authoritarian control. (Simon Davis, 1998)

> The Internet enjoyed by the West is a choice—not fate, not destiny, and not natural law. (Jack Goldsmith and Tim Wu, 2006)

In fact, the hierarchical structure of the Internet facilitates censorship and surveillance. Most traffic flows through a few backbone routers where filtering and monitoring is easily and cheaply applied. In many cases, these routers are government-owned, giving authorities full control over the traffic flowing through them. In the case of privately-owned routers, their concentration in the hands of a small number of communication corporations simplifies the installation of monitoring software as well. Coercing a small number of businesses into installing monitoring software is simpler—and more easily kept out of the common public knowledge—than a larger number.

A large body of work attempts to add censorship-resistance and surveillance to the Internet through a variety of, usually cryptographic, means. To date, such efforts have not seen wide-spread adoption, likely due to the difficulty of managing encryption keys [5] [26].

[5] Public key infrastructure has worked reasonably well for securing sensitive web transactions against arbitrary attackers, but is, at least in its current form, highly susceptible to government attack [25]. As a hierarchical system, a government can relatively easily coerce a widely-trusted root certificate issuer into signing an invalid key.

In this thesis, we instead propose non-hierarchical communication networks comprising the smartphones already carried by millions of people that we believe make wide-scale surveillance and censorship economically infeasible[6]. Such networks do not contain choke-points through which most traffic flows, so wide-spread censorship or surveillance would require controlling a substantial fraction of the devices or, in the case of wireless transmissions, monitoring a substantial fraction of the airspace. Eliminating hierarchy does have disadvantages—particularly in reduced bandwidth and increased latency—and the resulting networks cannot support all types of Internet traffic. But for many important applications, like text-based communication among friends and family, the network performance is acceptable.

1.1 Techniques for Combating Censorship and Surveillance in the Internet

Internet traffic is subject to observation and selective blocking, due to the hierarchical structure. Thus, methods to combat surveillance must obscure the information contained in the traffic, referred to as *privacy*, and the identities of those communicating, referred to as *anonymity*. Methods to combat censorship must further make the traffic indistinguishable from that which should not be blocked, e.g., economically-vital business transactions.

Privacy is usually maintained by encryption [27]. Protocols like SSL/TLS [28] are used for end-to-end encryption of general traffic streams, e.g., between a browser and a web server or an email client and an IMAP server. PGP [29] and OTR [30] provide encryption of emails or instant messages, respectively, between two parties, regardless of the intermediate transmission or storage protocols.

Anonymity is usually provided through variants of Chaum's mix-nets [31]. For example, mixminion [32] anonymous remailers provide anonymous email delivery. The Tor

[6] If not infeasible, at least much more expensive and much more visible to the public.

network [33] uses onion routing to provide anonymity for arbitrary TCP streams, including web traffic.

Others have noted that defeating censorship and surveillance does not always require full end-to-end privacy or anonymity. Those are only necessary while the traffic is flowing through the censor-controlled routers. For example, consider a user in Iran accessing a website critical of the Iranian government and hosted in the United States. Only while inside of Iran's network must the traffic be indistinguishable from other government-approved traffic. The recent Telex system [34] addresses traffic intended for a blocked service to uncensored services, but cryptographically tags the content such that trusted routers outside of the censored network can identify and redirect the packets to the intended destination.

Encryption is not a panacea, however, due to possible man-in-the-middle attacks and the resulting key distribution problem. A method of ensuring that the key used for encryption (public or secret) actually belongs to the intended recipient is needed. Most services, like SSL/TLS, use some form of public key infrastructure. Centralized authorities, whose public keys are distributed a priori and implicitly trusted, sign certificates linking identities to public keys. Clients needing to verify the public key for a particular party can validate the digital signature on the presented certificate. Unfortunately, the centralized nature of public key infrastructure makes it susceptible to government control. A centralized authority can intentionally issue false certificates due to coercion or unknowingly due to hacking [25].

Decentralized key distribution schemes are used as well. For example, PGP uses the web of trust. Individual users can sign each others' public key, attesting to their authenticity. Although one may directly trust a particular key, if it is signed by someone (or several people) that are trusted, one can choose to trust it. Systems like OTR [30] and ZRTP [35] do not require a prior key exchange, but instead use Diffie-Hellman key exchange [36] to establish a shared secret on first contact. OTR uses verification of arbitrary mutually-shared information to detect man-in-the-middle attacks and ZRTP uses voice-based verification of a shared value derived from the supposedly-shared key.

1.2 Advantages of Non-Hierarchical Networks

Non-hierarchical architectures have two primary advantages over their hierarchical counterparts.

Wide-scale censorship is nearly impossible, because it requires controlling a significant fraction of the participating devices.[7]

Similarly, spying on large fractions of the network traffic requires monitoring or controlling large portions of network and thus is difficult and likely prohibitively expensive. This property is particularly useful for some anonymity techniques, like onion routing, that are subject to traffic analysis attacks. Obtaining a broad view of the traffic patterns is much more difficult in the non-hierarchical network.

1.3 MANET Architectures for Communication

In this thesis, we consider a particular type of non-hierarchical network called a Mobile Ad Hoc Network (MANET). A MANET is a self-organizing network of mobile devices that communicate directly with nearby devices via wireless radio. Messages intended for recipients not in direct radio can hop through multiple devices to reach their destination. Such networks are inherently non-hierarchical, as all participants have essentially equal computational power, bandwidth, and range.

Most smartphones and latops are equipped with WiFi transceivers capable of ad hoc communication, making such networks an economically-feasible alternative to the Internet or cellular networks. These devices are already owned and used by a large number of people. Missing are the network protocols and software implementations needed to use the ad hoc capabilities for censorship and surveillance-resistant communication, a problem we begin to address in this thesis.

[7] Such control is theoretically possible, for example, by mandating that all devices come with special hardware or software, but this is disruptive, noticeable, and expensive to enforce.

MANETs are not without disadvantage. Bandwidth, latency, and energy consumption scale much more poorly than in infrastructure-based, hierarchical networks. Consequently, we focus on personal communication—email, text messaging, microblogging—whose throughput ($<$500 kbps) and latency (5–10 s) requirements are achievable.

1.4 Contributions and Organization

This thesis starts from the proposition that non-hierarchical network structures are inherently more resistant to censorship and surveillance than the hierarchical Internet[8] and thus should be designed and developed for real world use. Although promising from a censorship perspective, wireless ad hoc networks are known to have limited scalability due to contention of the wireless channel and the super-linear scaling of routing traffic. However, we show that these challenges are not insurmountable, as summarized by the following statement.

Thesis Statement: Non-hierarchical network architectures, which we believe are inherently more resistant to censorship and surveillance than the hierarchical Internet, can support common, useful, text-based communication applications, i.e., text-messaging and microblogging.

Towards this end, we propose architectures for two styles of communication, develop tools to address the difficulty of testing ad hoc networks with real users, and solve theoretical problems underlying the proposed architectures.

- In Chapter 2, we propose Whisper, a MANET architecture for secure and anonymous personal communication among friends and family. At the core of Whisper are two novel techniques: (1) a variant of onion-routing suitable for ad hoc networks, in which a set of onion routers is not available a priori, and (2) an efficient routing scheme based on the predictability of human motion. Chapter 5 solves a particular technical

[8] Aside from the arguments in this introduction, this thesis does not attempt to defend this claim. The arguments justify the contained research and development of non-hierarchical networks, but only time will tell if these intended benefits prove true in the real world.

problem, identifying non-Sybil identities in a one-hop neighborhood, required by the Whisper architecture.

- In Chapter 3, we propose Shout, a MANET architecture for censorship-resistant, Twitter-like, public microblogging. Shout uses manual human interaction to propagate messages, concentrating limited network bandwidth on messages of broad interest. Chapter 6 solves a particular technical problem, identifying spammers in a fully-decentralized network, needed for Shout to see widespread adoption.

- In Chapter 4, we describe MANES, a mobile ad hoc network emulation system designed to allow researchers to test their ad hoc networking protocols and applications with hundreds or thousands of real users by deploying them on standard Android smartphones. Most research on ad hoc network protocols is based on simulation or small-scale studies with tens of users, primarily due to the difficulty and cost of large-scale deployment. Both Whisper and Shout depend critically on behaviors of the underlying human users, so large-scale studies are needed. MANES emulates ad hoc connectivity over a cellular or Internet connection, and thus can be used with any Android phone without interfering existing WiFi usage. Further, it gives the researcher view of and control over the network.

- In Chapter 5, we describe the Mason test, a protocol for detecting Sybil attacks in wireless networks. The Whisper protocol requires participants to periodically gather sets of distinct neighboring identities, for later use in mix-chains. A neighbor conducting a Sybil attack, i.e., pretending to be multiple identities, would violate the distinctness requirement and potentially the security of later mix-chains. Noting that the received signal strengths of transmissions are hard to predict, the Mason test uses the *untrusted* RSSI observations reported by network participants to identify transmission originating from the same node.

- In Chapter 6, we characterize the user behavior in and the retweet graph of Twitter.

The resulting models are useful for driving simulation-based analysis and design of other microblogging systems. Implications of these results are discussed. As an example application, we develop a method for detecting spammers suitable for decentralized microblogging systems based the connectivity of the reshout graph. The identified properties of the retweet graph—scale-free and small-world—enable the generation of synthetic retweet graphs to evaluate the classification performance.

CHAPTER 2

Whisper

2.1 Introduction

Wireless mobile ad hoc networks (MANETs) composed of volunteer, mobile devices offer some advantages over traditional infrastructure networks because their nonhierarchical nature eliminates critical points of failure that can be exploited by attackers to reduce reliability and enable censorship, surveillance, and other forms of undesirable interference. Attacks upon communication systems are easier when most network traffic is routed through backbone networks owned by a few ISPs or a state [37]. MANETs have the potential to significantly increase the cost of large-scale censorship or shutdowns. Unfortunately, communication and computation capacities of individual nodes limit scalability [38] and have, thus far, undermined general-purpose use. However, use in specific applications remains a possibility. In particular, while MANET bandwidths and end-to-end latencies may be insufficient to support voice conversations or video, they may support valuable services like text messaging.

2.1.1 MANETs May Offer A More Robust Supplement to the Internet

> The Net interprets censorship as damage and routes around it. — *John Gilmore, 1993*

Although the Internet has been heralded for being robust to censorship, ongoing events in the Middle East, North Africa, Asia, and elsewhere falsify this belief; governments can exploit the hierarchical nature of the Internet to censor news as well as limit and monitor communication. In an extreme example, Egypt completely disabled Internet access for several days in February 2011 by forcing their five major ISPs to withdraw Border Gateway Protocol routes [39]. In Tunisia, where bandwidth is leased from the government [15], Internet access is heavily filtered. Many websites (e.g., YouTube) are blocked [15]. Others (e.g, Facebook and Twitter) are modified to steal login credentials [16]. Emails and attachments are filtered and scrubbed [15]. In all these cases, the choke-points inherent to the Internet's hierarchical structure help facilitate the censorship.

In contrast, mobile ad hoc networks composed of volunteer, wireless devices (e.g., smartphones and laptops) have the potential to be more resistant to corruption. Due to their nonhierarchical, ad hoc structures, censoring communication requires controlling many of the nodes in the network. When these nodes are handheld devices owned by private individuals numbering in the tens of thousands or more, acquiring such control is vastly more difficult and expensive than adding filtering software to a few backbone routers. Although MANETs will not help for long-distance or transocean communication, they have the potential to provide secure and uncensored communication within contiguously populated local regions, which may be sufficient to support communication among friends and family members.

2.1.2 MANET Architectures Should Exploit Application-Specific Properties

An ideal robust supplement to the Internet would support all types of traffic. Unfortunately, poor MANET scalability precludes their use for general-purpose networking. Thus, instead of seeking a general MANET architecture, we argue that MANET architectures must be tailored towards specific application-classes.

This poor scalability stems from two primary properties. (1) The traffic forwarded by each node increases with network size, reducing throughput for originating traffic [38]. (2) The traffic required to maintain routing state for the mobile nodes increases with network size, reducing available bandwidth [40]. Simulations indicate that current MANETs scale to only a few thousand nodes, with low per-node throughput ($<$5 kbps) [41].

We argue that these limitations imply that useful MANET architectures must be tailored to specific application-classes. First, the throughput and latency induced by the required network size must be acceptable. Second, properties of the application should be leveraged to design more efficient routing methods. In this work, we use predicted human motion patterns to support a MANET for text-based personal communication (e.g., text messaging), a low-bandwidth and latency-tolerant application.

2.1.3 Background on MANET Connectivity

The architecture described in this chapter requires good connectivity in the underlying MANET. In this section, we briefly review results from Bettstetter [42] that give the node densities for the network to be (probabilistically) connected.

Assume the participating devices are uniform randomly distributed over some geographic area and each has a transmission range of 100 m. We would like to know the density of devices required for the network to be connected and thus permit communication between arbitrary devices. Bettstetter [42] derived analytical expressions for such densities. Specifically, with a device density of 538 km^{-2}, the network is connected with 99.9% probability. A connected network is not necessarily robust—the removal of a single device might break the connectivity—so one can also consider the k-connectivity, where k is the number of devices that must be removed to break connectivity. Here, a device density of 904 km^{-2} gives a 99.9% chance of being 5-connected. Finally, the most relevant metric is the path probability—the probability that any pair of devices have a path between them and thus can communicate. Although analytical results are not available for this query, Bettstetter's

simulations indicate that a device density of 255 km^{-2} gives a path probability of 99%. Real people are not uniformly distributed—they cluster in places like rooms and buildings—so these densities can be viewed as lower bounds.

To put these densities in perspective, consider a college town like Ann Arbor, Michigan. Ann Arbor has a population density[1], of 1580 km^{-2}, suggesting that one-sixth of the population would need to participate in the network to achieve the 255 km^{-2} device density required for 99% path probabilities. As mentioned, people are not uniform randomly distributed, so higher participation would be required in areas of relatively low density to bridge the surrounding, presumably more dense, clusters. Although not conclusive (one can easily imagine networks partitioned by areas of low density, e.g., empty parking lots at night), these numbers indicate that town-scale ad hoc networks are feasible—the required device density and adoption rates are achievable.

2.1.4 MANET Architecture for Text-Based Personal Communication Applications

Text-based personal communication among friends and family members is both useful to many people (as evinced by the popularity of text and instant messaging) and particularly suited to a town-sized MANET, as indicated by the following two properties. (1) The required per-node throughput is low (<500 bps) and relatively high latency is acceptable (1–5 s). (2) People frequently communicate with relatively small groups of contacts in close geographic proximity [43], implying a short average link length, which improves scaling properties. Furthermore, properties of human motion patterns can be leveraged to provide efficient routing.

A MANET architecture supporting text-based personal communications should satisfy the following requirements.

- **Scalability**. A useful personal communication network must cover a region of non-

[1] According to the 2010 U.S. Census.

trivial area (e.g., a small town or a university campus), providing reliable delivery for all participants (e.g., a few thousand nodes) without imposing much computation or battery energy overhead on participating nodes. We require a per-node throughput on the order of 100 bps and delivery latencies on the order of 10 seconds.

- **Confidentiality**. The network should guarantee end-to-end message confidentiality. Packets should therefore be protected from eavesdropping and traffic analysis as they are relayed through arbitrary nodes untrusted by the source and destination.
- **Location Privacy**, defined as "the ability to prevent other parties from learning ones' current or past location" [44]. Persistent identifiers must not be linkable to node locations.
- **Social Network Privacy**. A person's social network, i.e., the set of network peers he communicates with, should be protected. No one (except the sender and receiver themselves) should be able to determine both the sender and receiver of any packet (by real identity, network identity, or location).

Meeting our scalability goals in a MANET is challenging because the route maintenance traffic required by typical routing algorithms quickly dominates total bandwidth and energy usage. On-demand protocols that reduce the load by delaying route-finding until necessary can provide constant-factor reduction, but do not change the scaling behavior. Stateless protocols try to eliminate maintenance traffic altogether by using only local information for, e.g., geographic location, for routing. However, this merely pushes the complexity and overhead into another domain, e.g., a distributed location service to map from node identities to geographic locations. An end-to-end routing method with reduced traffic overhead is needed.

In this chapter, we present the design of a location-centric MANET architecture supporting text-based personal communication within town-sized regions. Properties of human mobility patterns motivate a novel routing method, *location profile routing*. Geographic routing [40] is at the core of its scalability: next-hop selection requires only local knowl-

edge within one-hop neighborhoods. However, to address a message the sender needs to know the destination locations, which are traditionally provided by distributed location services [45] that scale poorly and do not easily support confidentiality and privacy. We observed that (1) humans have highly predictable motion patterns, spending the majority of time in a few locations [46] and (2) the frequency of change in mobility patterns is on the order of months and years. We propose to model location patterns as *location profiles* (e.g., location–probability pairs), distributing them face-to-face, instead of real locations via the network, to reduce overheads (see Section 2.2). Direct visibility of location profiles is often unacceptable, so we embed the pre-shared location profiles in encrypted reply blocks [31], thus preserving location privacy by hiding the destination from the sender (see Section 2.3). The reply blocks also provide sender–receiver unlinkability and public key encryption provides confidentiality (keys are shared along with the location profiles, so PKI is not necessary).

Note that our primary goal is providing a censorship-resistant communication system for day-to-day use, when human motion is highly routine and predictable. Our primary target is not Internet shutdowns in an active protest or revolution scenario (à la Egypt in February 2011) where movements may be highly varied and non-routine. However, our system still enables communication in these scenarios, with the scalability dependent on the extent that locations are predictable (e.g., when protesters are at home). Supporting such communication during protests is a secondary goal. Our primary goal is therefore supporting communication among friends and family members.

We make the following primary contributions[2].

- We propose leveraging the predictability of human motion to reduce routing costs in MANETs comprising handheld devices.
- We develop a reply block-based scheme to add location privacy to geographic-based

[2]This work was performed in close collaboration with several people. In particular, Yue Liu made crucial contributions to the design of the reply block technique for anonymity. Professor Robert P. Dick suggested the use of predictable motion patterns to reduce routing overhead and also helped design the reply block technique.

routing.

- We describe a location-centric MANET architecture that provides scalable and secure text-based personal communication that resists censorship and shutdown.

The rest of the chapter presents a detailed description and justification for this architecture. Section 2.2 presents location profile routing. Section 2.3 proposes the location reply blocks used to address the security and privacy issues induced by location profiling. With these two fundamental components developed, Section 2.4 describes the full scalable and secure location-centric network architecture.

2.2 Location Profile Routing

2.2.1 Introduction

Traditional routing protocols rely on the sharing of global information and thus scale poorly in mobile ad hoc networks (MANETs) with frequent changes in topology. Routing overhead grows quadratically in the number of nodes for distance vector and link state protocols [47] that must distribute changes to all nodes. The natural hierarchy used to reduce the overhead traffic in networks like the Internet (e.g., CIDR) is not available. On-demand methods [48–50] delay routing table updates until needed, but only reduce overhead by constant factors—the scaling behavior is unchanged.[3] Instead, stateless protocols that use local information to make forwarding decisions have the potential to scale.

One stateless protocol, Greedy Perimeter Stateless Routing (GPSR) [40], uses geography instead, requiring messages to be addressed to specific locations. Nodes already know their own locations (e.g., via GPS), allowing each intermediate step to bring the message closer to its destination. No global routing state is needed. Essentially though, this technique just shifts the complexity from routing to addressing. A forwarding node only needs its own

[3] We assume that sender and receiver locations are not correlated; that could change the scaling behavior.

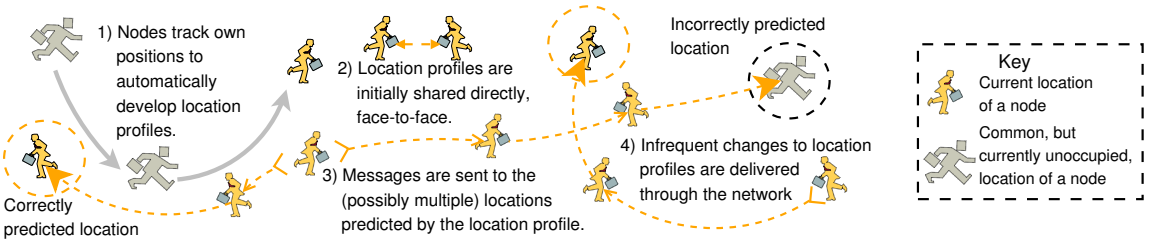


Figure 2.1: Illustration of the main components in location profile routing [1].

locally-known location, but the original sender requires the current location of the recipient, a global mapping.

Distributed location services [45, 51] can maintain this identity to location mapping, but also have drawbacks. Hierarchy is imposed to manage scalability, but overhead still increases super-linearly [45]. Further, locations are sensitive information, so complicated schemes are required to protect privacy and anonymity [44]. We observe that if node locations are predictable, the mapping can be done locally as well, reducing the scaling and privacy concerns.

In fact, human locations are highly regular with ∼93% predictability [52]. In MANETs of human-carried devices, predicative models of future locations can be pre-shared among trusted participants. These models combined with GPSR allow zero-overhead addressing and routing. Network scalability is limited by the actual traffic, not routing and location service overhead. We name this approach *location profile routing* (LPR) [1] and in this section study its performance potential. We determine the number of locations that must be addressed to achieve the peak 93% packet delivery rate and derive the associated latency and traffic overheads. Finally, we determine the conditions under which LPR outperforms GHLS.

2.2.2 Description of Location Profile Routing

Location profile routing (LPR) stems from the observation that humans generally have simple, repeated motions, spending most of their time at a few common time-dependent

locations [46] easily captured by a compact predictive model. For the many potential[4] applications of human-carried MANETs that can tolerate the resulting reduction in delivery reliability or increase in latency (we previously detailed a particularly compelling application—censorship-resistant personal communication [1]), LPR eliminates overhead traffic for route maintenance.

Figure 2.1 illustrates the main steps of LPR. Nodes continuously monitor their positions to build location profiles (step 1), which are then shared with potential future contacts directly (step 2). This sharing happens out-of-band, shielding the MANET from worst-case quadratic scaling behavior. A message is addressed to the location(s) predicted by the corresponding profile (step 3) and delivered via GPSR. Routing fails if a receiver is not in any of the predicted locations, but delay-tolerant delivery is a possible fallback. Changes to the motion patterns are rare (e.g., when someone starts a new job or moves to a new home) and can be distributed via the network (step 4).

Location Profiles: Motion patterns can be modeled in many ways, but a simple discrete model is sufficient for our purposes. A location profile is a function P mapping a time interval (e.g., Tuesday 15:30–15:40) to a set of location–confidence tuples, with higher confidence indicating stronger belief in the node occupying that location at that time:

$$P : \text{time} \mapsto \{(\text{loc}_1, \text{conf}_1), \ldots, (\text{loc}_n, \text{conf}_n)\}$$

The precise discretization level is unimportant. Both cell-tower granularity (3 km^2, 1 h) and WiFi AP granularity (157 m^2, 10 min) have similar predictabilities at 93% [52] and 92% [53].

Various implementations are possible, but for completeness we summarize the Prediction-by-Partial-Match (PPM) approach of Burbey and Martin [53], which is sufficient. PPM is a variable-order Markov model over a sequence S of observed time-interval–location pairs, $S = \{T_1 L_1 T_2 L_2 \ldots T_n L_n\}$. This defines a probability distribution over the next location

[4] Ad hoc networks are not yet widely used by the general public.

conditioned on the prior k elements of context. In our case, prior locations are not known, so our definition of P corresponds to the first-order variant ($k = 1$, i.e., context is the current time). We briefly discuss zero- (no context) and third-order (context includes the previous location) variants. This scheme captures most of the predictability (90% [53] vs. the 93% reported maximum [52]).

Profile Distribution: Location profiles are disseminated a-priori and out-of-band, similar to telephone numbers or email addresses. For our envisioned applications—communication between friends—the profiles can be exchanged face-to-face. In other cases, a centralized service, similar to a telephone directory, might be needed. Regardless, the salient point is that the profiles are known a-priori and thus can be exchanged outside of the network. Although changes could be disseminated out-of-band as well, in-network propagation is feasible because updates are infrequent and sent only to select participants (e.g., friends). Opportunistically updating when devices are in close proximity further bounds the overhead.

Addressing Policy: The addressing policy translates the location–confidence tuples output by the profile into a message delivery strategy specifying when and where packets will be sent. Only one of the locations can be correct, so the order and method in which they are tried influences the network throughput and latency trade-off. Their spatial correlations influence the minimum cost routing strategy (e.g., Steiner tree) to reach all locations. The primary focus of this paper is analyzing these performance characteristics and trade-offs.

Fallback Method: LPR fails outright when nodes are in unpredictable locations, i.e., at least 7% of the time [52]. Although this may be tolerable for many applications in which messages can be redelivered later, it is non-ideal. As this is not our focus, we omit details here, but possible strategies include delay tolerant delivery (in-network buffering of the message at a common location until the node's return) or rendezvous delivery (messages are sent to a rendezvous location which the node, when not in a predictable location, apprises of current forwarding instructions). Such schemes allow for reliable delivery with average overheads still drastically lower than traditional routing approaches.

2.2.3 Performance Analysis

We use prior empirical studies of human motion patterns to develop analytical models suitable for studying the performance of LPR. Barabási et al. studied six-month location traces of 100,000 European cellphone users [46, 52] at cell-tower granularity, reporting a maximum predictability of 93%. The size and duration of the traces make this best source to date. To confirm that locations are as predicable at WiFi granularity, we turn to Burbey and Martin's study [53] study of traces from 275 WiFi users at UCSD [54]. They found similar predictability, 92%, confirming that cellular granularity is not limiting.

2.2.3.1 How Predictable are Common Locations?

A location profile returns multiple locations in order of likelihood, so delivery cost and latency depends on how many, K, must be targeted to reach the user. Intuitively, most time is spent in two locations—home and work—so a zero-order model (i.e., not conditioned on current time) might be sufficient. The pmf is $\widetilde{\pi}(k) = p_k \prod_{i=1}^{k-1}(1-p_i)$, where p_i is the probability that the target is in location i. The p_i's are roughly distributed[5] as $p_i \propto i^{-1}$ with proportionality constant $c \approx 0.48$ [46]. K is equivalent to a beta-geometric distribution, $K \sim \mathrm{Geom}(L)$ with $L \sim \mathrm{Beta}(c, 1-c)$, and has CDF

$$\widetilde{\Pi}(k) = 1 - \frac{1}{k\,\mathrm{B}(k, 1-c)}. \tag{2.1}$$

The match[6] to measured data [52] is shown in Figure 2.2. The first moment diverges, but two locations suffice only 60% of the time and ten achieve only 80% delivery. Conditioning the model on time of day is necessary.

The first-order model (with 10 min intervals) is 90% accurate for the first location on

[5] A true Zipfian distribution requires a bounded domain $i \in [1, N]$ with $c = \frac{1}{H_N}$ for the p_i's to total one. The following results are for the reported empirical form, not a true distribution.

[6] $L \sim \mathrm{Beta}(0.60, 0.72)$ yields a much tighter fit, but we lack an explanatory origin. It *might* result from a mixture of different upper bounds N in the Zipfian model of the p_i's—individuals have different numbers of common locations.

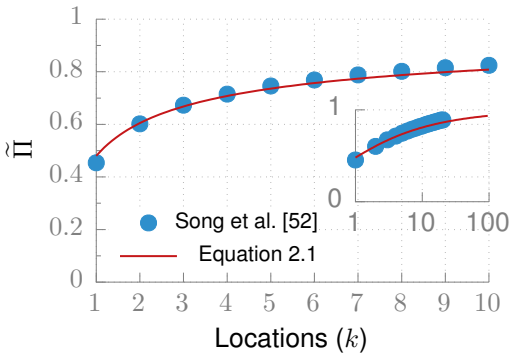


Figure 2.2: The probability that a user currently occupies one of his k most-common locations is well-modeled by Equation 2.1.

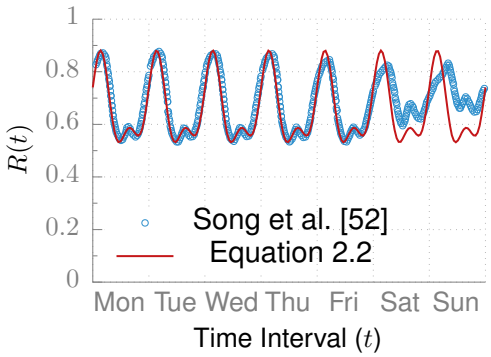


Figure 2.3: The time-dependent regularity $R(t)$, i.e., the probability the user is in the most common location associated with that time interval.

the UCSD dataset [53], nearing the 93% upper bound and suggesting marginal gains for additional guesses. A third-order model is surprisingly only slightly better at 92%. The larger cellular dataset (with 1 h intervals) is more pessimistic. The accuracy $R(t)$ of the first-order model here is given by

$$R(t) = c_1 \sin\left(\frac{2\pi}{24}t + \frac{2\pi}{8}\right) + c_2 \sin\left(\frac{2\pi}{12}t - \frac{2\pi}{24}\right) + c_3, \tag{2.2}$$

where $c_1 = 0.148$, $c_2 = 0.077$, $c_3 = 0.657$ and $t \in [0, 167]$ is the hour of the week, i.e., $t = 0$ is Monday 00:00–0:59 and $t = 167$ is Sunday 23:00–23:59. As shown in Figure 2.3, this form captures one-day and half-day periodicities. On weekends, the variability is lower and the intervals of highest predictability occur later in the day The accuracy on weekdays ranges from 55% to 90%, averaging $\bar{R} \approx 65\%$.

Assuming the power law form, $p_i \propto i^{-1}$, holds during each time interval[7], equations 2.1 and 2.2 can be combined as

$$\widetilde{\Pi}_1(k) = 1 - \int_0^{168} \frac{D(t)}{k\,\mathrm{B}(k, 1 - R(t))}\,dt, \tag{2.3}$$

[7]The number of common locations is inversely correlated with $R(t)$ [52] (Fig. 3B), suggesting that it does.

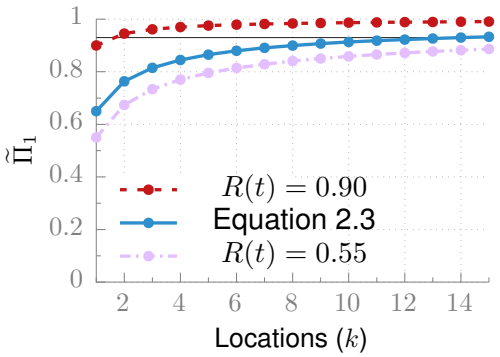


Figure 2.4: Success rate of a first-order profile versus the number of locations attempted. Rates during maximum (night) and minimum (day) predictability are shown too.

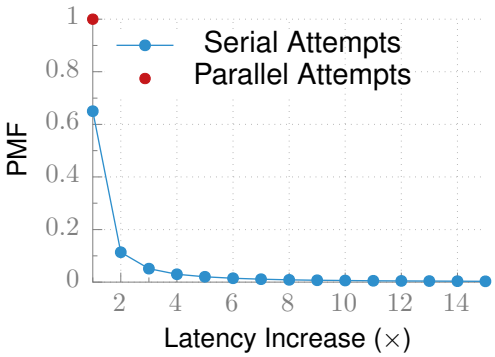


Figure 2.5: PMF of the latency increase for the first packet in a stream induced by trying multiple locations in turn. Concurrent attempts do not impact latency.

where $D(t)$ is the traffic density at time t, to yield the average probability that packet addressed to the k-most common locations reaches the target, shown in Figure 2.4. We assume a uniform density, $D(t) = \frac{1}{168}$, but other known traffic patterns can be substituted. $k = 5$ achieves 85% success and 93% requires only $k = 12$. More locations are required during the day and fewer at night. The exact number of locations to attempt is application-specific, depending on the trade-off between between desired delivery rate and cost, i.e., increased latency and traffic overhead.

2.2.3.2 What Additional Latency and Traffic is Induced By LPR?

Some packets must be sent to multiple locations to have an adequate packet delivery rate, increasing latency and traffic by constant factors. Note that the costs increase only for the first packet in a stream. Subsequent packets are sent directly to the now-known current location. The true average overhead depends on the percentage of first packets, which is low for applications like text-messaging and email and higher for interactive applications like voice chat. We report overheads for first packets only, which readers should scale by the first packet percentage of their applications.

We assume that receiver common locations and sender locations are uniformly distributed

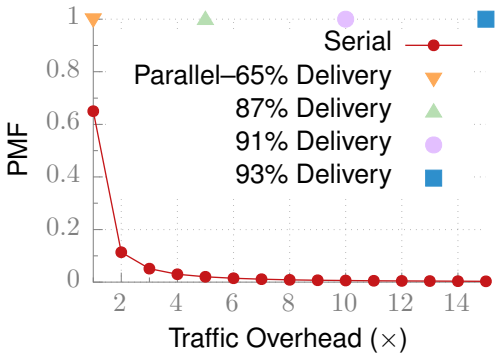


Figure 2.6: PMF of the traffic overhead for the first packet in a stream induced by trying locations in turn. Concurrent attempts have a fixed overhead.

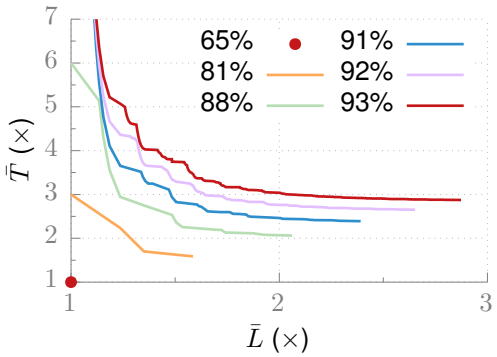


Figure 2.7: Pareto front of the first packet latency–traffic trade-off of a combined parallel-series strategy for several average success rates.

in the network.[8] Thus, we can report overheads relative to the average latency (round-trip time) and traffic cost (round-trip hop count) for a single delivery attempt, e.g., a $2\times$ increase.

Parallel delivery to all k common locations does not increase latency, but increases traffic k times. Serial delivery—attempting each location only if the previous failed, using ACKs and a timeout to determine success and failure—reduces the traffic overhead. The pmf of the factor increase T, plotted in Figure 2.6, is

$$\Pr[T = t] = \tilde{\pi}_1(t), \tag{2.4}$$

where $\tilde{\pi}_1$ is the pmf associated with Equation 2.3. Latency increases similarly, as shown in Figure 2.5.

A combined approach—addressing a subset of the locations in parallel—can fine-tune the trade-off. For example, four different groupings can be used when trying three locations ($\sim 81\%$ success rate).

$$\boxed{1\ \ 2\ \ 3} \qquad \boxed{1\ \ 2}\,\boxed{3} \qquad \boxed{1}\,\boxed{2\ \ 3} \qquad \boxed{1}\,\boxed{2}\,\boxed{3}$$

All locations within a group (a box in the diagram) are tried concurrently and groups are tried serially from left to right, as needed. Formally, a grouping G is a partition of the common

[8] Overheads can be lower if locations are spatially correlated, we discuss in later sections.

locations, $G = \{g_1, g_2, \ldots\}$, with the property that for $i < j$, all locations in group g_i are more probable than those in g_j. Let $\kappa(g)$ denote the index of the most common location in g, e.g., $\kappa(g_1) = 1$. Then, the probability that group g is tried, i.e., that all previous groups failed, is $\Phi_1(g) = 1 - \tilde{\Pi}_1(\kappa(g) - 1)$. Thus, the average latency increase $\bar{L}$ for a grouping G is given by

$$\bar{L}(G) = \sum_{g \in G} \Phi_1(g) \tag{2.5}$$

and the average traffic overhead $\bar{T}$ by

$$\bar{T}(G) = \sum_{g \in G} |g| \Phi_1(g). \tag{2.6}$$

Figure 2.7 shows the Pareto fronts for several average success rates, i.e., the maximum number of locations attempted. At the knees, $\bar{L} \approx 1.25\times$ and $\bar{T} \approx 3$–$4\times$. These curves are network averages. At runtime when a specific location profile is known, the precise trade-offs for that instance can be computed.

2.2.3.3 Under What Conditions Does LPR Outperform Location Services?

LPR trades the cost of updating a location service as devices move for multiple transmissions at the first packet. We use a simple analytical model to derive the network conditions under which LPR outperforms GHLS [45], a scalable distributed location service. Let f be the network-wide location update rate (increases with node movement), r be the network-wide first-packet rate, s be the average number of hops between a node and its GHLS location server, p be the average number of hops between a source and destination, and $\bar{T}$, as previously defined, be the average number of destinations attempted by LPR. The location update, location query, and first-packet delivery costs (i.e., transmission counts) for GHLS are[9] fs, $2rs$, and $2rp$. LPR has only the first-packet delivery cost, $2\bar{T}rp$. After rearranging

[9] See Das et al. (Section IV) [45] for the derivation. Our s is their $\frac{1}{3}2^{h-1}\sqrt{2}$.

the total costs in terms of $\frac{f}{r}$ and $\frac{p}{s}$, we see that LPR has lower overhead when

$$\frac{f}{r} > \frac{p}{s}(2\bar{T} - 2) - 2. \tag{2.7}$$

When $s = p$ (source and destination are uniformly distributed over the entire field) and $\bar{T} \approx 3$ (from Figure 2.7), this simplifies to $\frac{f}{r} > 2$; LPR outperforms GHLS when the location update rate is more than twice the first-packet rate. This bound further decreases when sources and destinations are spatially concentrated, i.e., $p < s$.

2.2.3.4 Reducing Overhead Via Spanning Trees

The preceding overhead and latency analysis assumed linear routing, i.e., one transmission from the source per attempted destination. A branching route (e.g., the Euclidean Steiner tree containing the source and destinations) would reduce this overhead, particularly when destinations are spatially-clustered relative to the source. Unfortunately, this works only for dense networks in which nodes are guaranteed to exist at the branching (e.g., Steiner) points. Many real-world networks are too irregular, and the linear approach should be used.

In dense networks, the branching approach is feasible. One desires a routing tree with low total weight to minimize traffic but also with short source-to-destination path lengths to minimize latency. Although seemingly conflicting, both goals are achievable. Taking n as the size of the network, trees with weights within $o(n)$ of the $O(n)$-length minimal Steiner tree and source-to-destination path lengths within $o(\log n)$ of the $O(\sqrt{n})$ straight-line distances exist [55]. We refer the reader to Aldous and Kendall for details and construction [55].

2.3 Privacy and Anonymity

MANETs are open to untrusted observation and participation, inducing several security concerns, e.g., location and social network privacy. Furthermore, our proposed routing scheme at first appears to require that users trust their contacts enough to share location

profiles, selectively giving up location privacy. Although this might be acceptable in some applications (e.g., when one's contacts already know the motion patterns), often it is undesirable. We propose a reply-block- and pseudonym-based scheme that enables location profile routing to operate without exposing location profiles (or identifying information), even to contacts. In this section, we define the desired security properties and describe our solution.

2.3.1 Attack and Trust Model

We assume that the attackers, in addition to participating, can observe all links in the network and collaborate using side-channels. They may have storage and processing capabilities exceeding those of a typical handheld device, allowing for traffic analysis of accumulated observations, and may triangulate the position of transmitters. We do restrict their number, assuming that economics dictates that conforming nodes will generally outnumber attacking nodes.

We do not consider attacks using information from outside of our protocols, e.g., taking photos of the human carrying a node for later identification. Similarly, we assume the other protocol layers, e.g., physical and application, are secure (as defined in Subsection 2.3.2). For example, wireless transmissions should not contain identifying analog "fingerprints" that would allow a node to be tracked. Of course, full system security requires that all layers have these security properties, but such provision is orthogonal across layers, so this work focuses on the network layer. Finally, we assume the majority of nodes obey our protocols, thus resisting routing attacks. We plan to quantify this resistance in future work.

We assume that the hardware and software platform, e.g., a smartphone and its operating system, is trusted, i.e., it is not modified to specifically interfere with our protocols or spy on the information transmitted. Although the hardware or software could be theoretically be modified, China's failed efforts to require the distribution of its Green Dam Youth Filtering software with new computers, a much simpler approach, illustrates the practical difficulty of

doing so.

We also assume that the Whisper implementation is uncompromised, e.g., it does not contain backdoors. On the assumption that servers are more easily compromised than many individual devices, we envision phone-to-phone distribution of the software, instead of downloads from a public server, to reduce the likelihood of compromised installations.

2.3.2 Desired Anonymity and Privacy Properties

The trust concerns in MANETs are often addressed by listing specific requirements for privacy (the confidentiality of information) and anonymity (the confidentiality of the relationship between an identity and its information, i.e., attributes or actions). We believe this approach has two primary flaws. First, it focuses attention on the security provided, when the security *not provided* is of greater importance and interest. Second, it suggests a false separation between the attributes of an entity and its identity. In reality, the attributes themselves often allow identification (e.g., the Netflix dataset fiasco [56]), so separating them from a "traditional" identity (e.g., a name or social security number) is false protection. Further, predicting which attributes could, in the hands of a clever-enough attacker, allow identification is difficult. Therefore, we adopt a methodology that puts focus on the security not provided and endeavors to provide complete anonymity, removing the need to attempt to accurately distinguish identifying and non-identifying attributes.

We focus attention on the unprovided security by starting from an unrealistically strong, but easy-to-define, security goal, and relaxing it by describing specific security properties that it implies but we cannot yet provide. These relaxations have two sources. Type 1 relaxations are inherent to the underlying implementation technology (e.g., with wireless communication technology, the location of the transmitter of a packet is always linkable to the packet). These cannot be considered flaws of our protocols and must be accepted. Type 2 relaxations are those induced by our protocols (e.g., we employ per-location pseudonyms to prevent tracking a node across space, but it remains possible to track a node in one

location, across time). These are clearly flaws of our protocols and are opportunities for future improvement.

We term our unrealistically strong starting point *complete anonymity* and use it to address the false separation of identity and attributes. Put simply, complete anonymity requires that each observable attribute in the network (i.e., the act of transmission and each data attribute within) be unlinkable to the other attributes from the same entity (i.e., node). More precisely, in a network comprising n nodes, an observer should have belief $\frac{1}{n}$ for each node that a given attribute originated from that node. Equivalently, for any two attributes, an observer will have an equal belief in their originators being the same node or different nodes. This strong unlinkability requirement prevents the inference of identifying information. For example, network participation is anonymous because an identifier (a set of data attributes) is unlinkable to transmission (an action).

In MANETS, we can decompose complete anonymity into six unlinkability relationships over three attributes: actions (e.g., packet transmission), traditional identifiers (e.g., MAC address, name, pseudonym), and locations. The following list summarizes these six relationships, describing the Type 1 and 2 relaxations:

action–location: In a MANET, transmission location is obviously visible, so this link is an allowed Type 1 relaxation. Actions must still be unlinkable to past or future locations of their entity.

action–identifier: Our protocols (given in the following subsection) use visible per-location pseudonyms, resulting in a Type 2 relaxation: each action is linkable to exactly one pseudonym. Actions must still be unlinkable to other identifier types.

action–action: Action–location linkability induces a slight Type 1 relaxation: two actions linkable to the same location can be linked. However, two actions at different locations must still be unlinkable.

identifier–location: An identifier should not be linkable to the location of its entity. Again, our solution using per-location pseudonyms will violate this slightly, resulting in a Type 2

relaxation: each pseudonym is linkable to exactly one location of its entity.

identifier–identifier: Two identifiers for the same entity should not be linkable. For example, a pseudonym must be unlinkable to other identities (e.g., real name) and multiple pseudonyms for an entity must be unlinkable. For network transmissions, this means that the identifiers of the sender and receiver cannot be linked, providing *social network privacy*. For personal communication, contacts know each other, resulting in one Type I relaxation: communicating contacts can link each other's identifiers. Pseudonyms induce one Type 2 relaxation: the pseudonyms of the sender and receiver for a one-hop (i.e., forwarding) transmission are linkable.

location–location: The current, past, and future locations of a node must be unlinkable. We allow one Type 2 relaxation: the allowed identifier–location link for per-location pseudonyms implies that past and future locations can be linked, but only when they are the same location. Critically, this provides *location privacy*, with the exception that the existence of a node at a single location may be tracked across time.

2.3.3 Unlinkability via Reply Blocks and Pseudonyms

This section presents our reply block- and pseudonym-based solution to provide the desired unlinkability. A formal argument that the properties are satisfied requires complete enumeration of all types of actions, identifiers, and locations and lengthy analysis. Such detail is beyond the scope of this dissertation , so instead the solutions are presented with high-level arguments for their correctness. Roughly, the following arguments derive from the premise that two attributes are unlinkable if (1) they are never both available in the same context and (2) transitive application of known relationships cannot be used to link them.

Geographic routing lends itself to our unlinkability requirements, because messages are addressed to locations, not identifiers. Identifiers are not visible in packet headers and thus the three identifier relationships are implicitly unlinkable by third parties. The

sender and receiver themselves do know each other's identifiers, so we use reply blocks, a variant of Chaum's *mix-nets*, to disassociate information available at the sender (receiver's identifier) from the information available at the receiver (receiver's location and receiver's actions) and vice versa, explicitly protecting the identifier relationships[10]. A reply block is a routing instruction that guides a message from a sender through a mix-chain leading to the receiver. A mix-chain is composed of *mix-servers*, each of which disassociates the incoming and outgoing messages by reordering them and changing their appearance via layered decryption. Thus, observers (including the sender itself) cannot track the original message; at any point, only the previous and next mix-servers are known. We give detailed descriptions of applying reply block techniques in MANETs, including how senders choose the mix-servers composing a chain, in the remaining parts of this subsection.

Action–action links are also protected. This linking would require transitive application of other relationships: action A linked to X and action B linked to X implies A is linked to B. Aside from the allowed Type 1 exception when X is a location, no such X exists; the action–identifier and identifier–location relationships are unlinkable.

Location–location links are also protected. The location caches shared with a contact are encapsulated in reply blocks, so the actual locations are not revealed to the contact. Further, as with the action-action link, transitive linking of locations is not possible: the mix-chain dissociates locations from other attributes.

Location-based addressing has one significant problem. The predicted locations are inherently imprecise, so messages must be addressed to relatively large regions (several 802.11b hops in radius) and then flooded, wasting significant bandwidth and energy. To address this, we introduce pseudonyms as secondary addresses. Messages are addressed to both a location and a pseudonym (both encapsulated in the reply block), with the location used for initial routing and the pseudonym used in the destination region. Different pseudonyms are used in each location, preventing the pseudonyms from transitively linking other attributes.

[10]The usual caveats for mix-chains apply. Linking is possible if all nodes in the chain collaborate and global traffic analysis can potentially reveal message flows in some special circumstances.

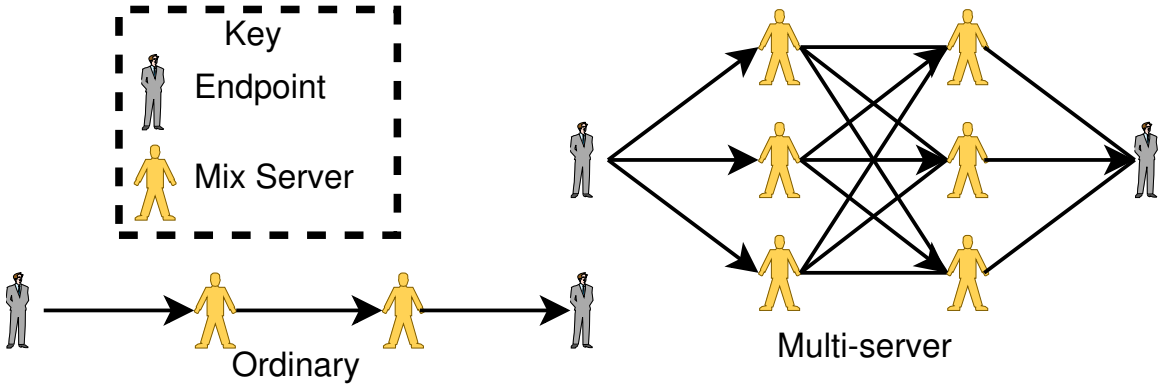


Figure 2.8: Message flow for ordinary and multi-server reply blocks.

However, they still violate the strictest requirements, resulting in the previously mentioned Type 2 relaxations. Three of these, action–pseudonym, pseudonym–location, and, for one-hop sender–receiver links, pseudonym–pseudonym, are acceptable because the pseudonyms map one-to-one to an already visible attribute, location, and contain no additional useful information. The fourth though, is unfortunate. Pseudonyms persist across time and can be used to link the times when a node is in the same location (a type of location–location link). We are investigating possible remedies. An obvious possibility is frequently changing pseudonyms.

2.3.3.1 Reply Block Operation and Management

The chain in Figure 2.8 illustrates the use of an ordinary reply block, specifying a two-server mix-chain. Each transmission depends on those before, posing a deliverability problem. Each mix-server provides a single common location, so with non-negligible probability the server will be unreachable at the time of attempted contact. We solve this problem by specifying multiple mix servers at each layer (also in Figure 2.8), increasing the probability of successful delivery. Each layer of the reply block is encrypted to three servers, who each remove an encryption layer and each forward the packet to the next three mix-nodes. Each server remembers the previous–next hop association. The receiver sends a message back through the fastest chain to complete, marking it as available for subsequent packets.

Reply blocks are location profiles anonymized by mix-chains, so managing them includes two main tasks: location profile management and mix-pool management. Each device needs to track its motions and keep its location profile up to date. Additionally, the mix-servers used in one's reply blocks also need to be valid. When there are significant changes in a device's location profile, or there are too many unreachable mix-servers in a reply block to permit any valid route, the reply blocks need to be updated accordingly.

2.3.3.2 Mix-Server Pool Management

Mix-server selection is important because if all mix-servers in a chain collaborate on an attack, the sender and receiver can be linked. Two selection requirements need to be satisfied. (1) Servers should have high probability of protocol compliance, reducing the chance that all servers in a chain improperly collaborate to trace a message. (2) They must be directly reachable by locations, instead of reply blocks, to prevent an infinite chain of reply blocks. For traditional Internet mix-chains, services are chosen from semi-trusted published lists, as with Tor [33]. However, this method is not suitable for MANETs; no semi-trusted authority who could publish such a list exists. A new method for choosing mix-servers is needed.

We assume that physical attacker density is limited by economic constraints, and thus propose that each node individually maintain a pool of mix-servers chosen randomly from the various one-hop neighbors it encounters. This density assumption could be violated by Sybil attacks [57], in which one device pretends to be many, so we develop a technique leveraging signal strength measurements to detect Sybil identities during pool population (Chapter 5). As a node moves in the network, it asks one-hop neighbors to act as future potential mix-servers. Willing neighbors respond with a single $\langle$common location, pseudonym$\rangle$ address and an associated contact probability. Entire profiles are not shared to preserve location privacy. The requester saves the information from non-Sybil neighbors in its mix-server pool for future usage.

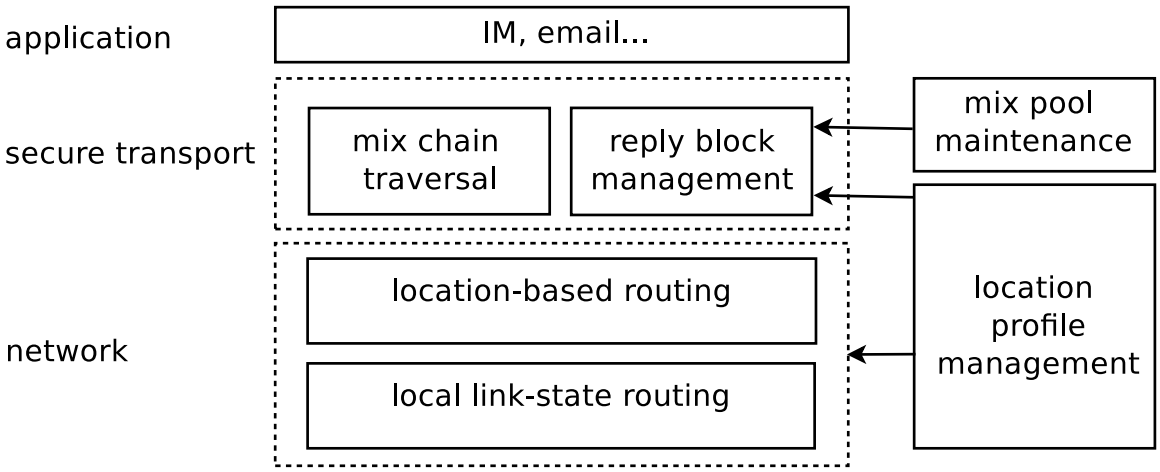


Figure 2.9: Main components of the location-centric network, with arrows representing service relationships.

2.4 Location-Centric Network

Now that the two most important pieces—location profile routing (see Section 2.2) and reply block-based privacy (see Section 2.3)—have been described, we present the architecture of our location-centric network for secure personal communication. System scalability relies on location-profile routing, into which we incorporate confidentiality and privacy mechanisms. As illustrated in Figure 2.9, the system comprises three layers, (1) application, (2) secure transport, and (3) network. The target application is low-bandwidth and delay-tolerant text-based communication, e.g., email and text messaging. The secure transport layer provides confidential and anonymous host-to-host delivery using mix-chains. The reply blocks constructed by a host and shared during face-to-face contact act as the transport layer addresses. The network layer delivers messages between mix-nodes using geographic routing. A network address is a two-tuple containing a pseudonym and location. Keys for encryption are exchanged face-to-face between contacts, so no PKI is required. In this section, we will describe the network and secure transport layers in more detail.

Network. Geographic routing (e.g., GPSR [40]) is the backbone of the network, providing routing scalability. Location profiles are exchanged face-to-face, providing location-distribution scalability, normally the Achilles' heel of geographic routing. A node's move-

ment within a small region prevents addressing destinations by precise coordinates, so we propose using geographic routing for coarse delivery and reactive routing near the receiver. Thus, a receiver is addressed by both a destination region and a pseudonym. When a message reaches its destination region, the intermediate node at the boundary transitions from geographic routing to local link-state routing. If a route is known, the message is delivered along it. Otherwise, a route discovery message is broadcast to discover one. If the node is unreachable, the message is dropped.

Secure transport. The transport layer provides host-to-host secure communication channels. A channel is a mix-chain between the sender and the receiver, constructed according to the receiver's reply block, that provides the desired location privacy and sender–receiver unlinkability. It is constructed according to the receiver's reply blocks. End-to-end encryption provides confidentiality.

We now describe the operation of the transport layer, responsible for delivering messages from the application layer to the destination node. To deliver a message, the sender first determines whether a channel to the destination is already available. If so, the message is sent via the channel. Otherwise, the sender sends a setup message using the receiver reply block with the highest contact probability. If the sender does not receive a response within a constrained time, it concludes that the receiver is not at the corresponding location of that reply block and repeats this process for the remaining reply blocks, until a response is received or all the reply blocks have been used. Our preliminary analysis indicates that, on average, receivers will be contacted via a reply block 93% of the time. When a response is received, the sender marks the channel as valid, sets a timeout for it, and messages are delivered thorough this channel. The receiver can respond via this channel as well, although the routing is not, in general, symmetric. Messages are encrypted with a session key established during the channel setup process.

Overhead. Energy consumption is a significant concern, especially since much of the work is forwarding others' traffic and does not directly benefit the user paying the cost.

Current 802.11 ad hoc technology is inefficient, depleting cellphone batteries in several hours (the power save mode is only for AP networks). Implementing a periodic sleep option for ad hoc mode will be necessary. Even with reasonable battery life, some selfish users might refuse to forward traffic for others, but we believe they will be in the minority. Most people derive some satisfaction from helping others, particularly at low cost (e.g., charging ones' phone each night instead of every other). An application feature displaying statistics about the number of conversations relayed could encourage such altruism.

2.5 Conclusion

We have not implemented the architecture laid out in this chapter. Instead, it's constituent components motivate two specific areas of research. We hope this architecture (or something similar) will be implemented, but the underlying pieces must be studied and developed first. Specifically, it motivates two primary research directions.

Human Mobility Patterns: Predictability motion patterns can substantially improve routing efficiency, but by how much? How predictable are human locations at the spatial granularity of a WiFi transmission range? How large is a predictable location and how many are non-trivially probable at a given time-of-day?

The data required to answer these questions—long duration traces of high spatial and temporal granularity for many people—are hard to obtain. We believe some of the data collected by the MANES system (Chapter 4) used to deploy Shout (Chapter 3) will be a useful starting point and are investigating it now.

Mix Node Selection: Safe reply block construction requires choosing multiple, distinct, anonymous participants chosen from among volunteering nodes. As mentioned, this is easily defeated by a Sybil attack, in which one device pretends to be multiple participants. Chapter 5, describes our defense against such attacks, the Mason test.

CHAPTER 3

Shout

3.1 Introduction

Usage statistics of services like Twitter and Weibo indicate the popularity and growing importance of microblogging communication applications. In 2012, Twitter had over 200 million active global users [58, 59] generating over 400 million tweets per day. In China, where Twitter is blocked by the government, the approved alternative, Weibo, reported over 36.5 million active daily users [60]. These short-form, public broadcasts have become a natural part of daily communication for many people worldwide.

In places where traditional media sources are heavily censored or controlled, social media has offered an excellent avenue for dissidents to educate and organize the general populace. The 2011 Arab Spring uprisings demonstrated this value, with Twitter used to share criticisms of the existing regimes in Tunisia and Egypt, sparking increased political debate and participation [61]. Spikes in online activity preceding protests indicated its usefulness in mobilizing large numbers of people and continued activity proved its ability to report from the action [61]. In China, Weibo is used by amateur reporters to great effect, raising public awareness of issues ranging from food safety to the extravagant lifestyles of government officials [62]. That content can be submitted by anyone and is filtered and judged by the general audience, not a select government official or media executive, is the real power of these services. Posts of widespread interest or importance quickly reach many

people.

Naturally, oppressive governments have responded by banning and censoring such services. In China, foreign services like Twitter are blocked, with domestic alternatives supporting government-specified censorship demands, like Weibo, appeasing public demand. Weibo appears to use a variety of censorship methods, including deleting posts containing banned keywords, rejecting sensitive search queries, and banning some users [10]. Various Arab governments similarly block access to Twitter [12, 13]. In response to protests in late January 2011, the Egyptian government, which had not previously blocked such social media sites, did so [14]. When that failed to stem the tide, all Internet access within the country was disabled for several days, after the major ISPs were forced to withdraw their Border Gateway Protocol routes [17, 18]. Given the demonstrated value of these microblogging applications in affecting social awareness and change, methods for resisting such censorship are needed.

Traditional censorship countermeasures like proxies [63, 64] and anonymous overlay networks [33] are not ideal for microblogging applications. In particular, they (1) still route all traffic through a few centralized chokepoints—the government- or ISP-owned routers—facilitating advanced traffic analysis[1], (2) require some level of technical sophistication for installation and operation, impeding widespread deployment, and (3) are easily defeated by blocking all external Internet traffic[2]. In this chapter, we instead argue instead for a microblogging architecture based on ad hoc networking, which are much more difficult to censor or surveil than the hierarchical, infrastructure-based Internet.

Certain properties of the microblogging communication style, particularly for sensitive content likely to trigger censors and incite government response, suggest the suitability of ad hoc networks.

[1] Although advanced traffic analysis does yet appear to be in wide use, its future use is likely as this cat-and-mouse game between censors and their targets continues.

[2] Since such shutdowns are economically damaging and thus likely to be short, the loss of social media access could be tolerated. But having a working Twitter-like system dissemination and organization is still preferable.

- Content is deliberately public. Activists intend to inform and organize broad portions of the public, not privately chat amongst themselves. Posts are intentionally visible to everyone, even officials of the protested government.

- The target audience is geographically dense, i.e., concentrated within a city or town. Whether organizing a demonstration in a public square or spreading facts about a corrupt politician, it's most critical that messages reach those nearby[3].

- High delivery latencies, on the order of minutes or hours, are acceptable. Microblogging is largely a distribution mechanism, not an avenue for interactive, back-and-forth discussion or debate. Much interesting content is relevant for several hours or days, so immediate delivery is not necessary.

- Content size is small. 1500 tweets—that's more than one per minute for an entire day—consume fewer than 500 kilobytes. At volumes reasonable for human consumption, microblogging requires very little bandwidth.

We propose Shout, a decentralized, ad hoc network-based architecture for microblogging designed to be difficult to censor. Shouts (tweets) and reshouts (retweets) are sent to neighbors within the one-hop broadcast range, flowing via the geographic, rather than social, network. This flow fits the intended distribution for censorable content well and the nature of the application tolerates the inherent bandwidth and latency limits of ad hoc networks. The non-hierarchical network structure is free of choke-points where censorship could be easily applied.

Our design has several unique aspects compared to traditional microblogging applications and ad hoc network protocols that greatly reduce system complexity and improve operation efficiency.

[3]We do not discount the importance of communication with the outside world. The 2011 uprisings proved this use of Twitter as well [61]. However, only a small number of links are needed to spread content between these separated clusters, e.g., cities or countries. Tech-savvy users comfortable with traditional proxies, anonymity services like Tor [33], or alternatives like *Speak-to-Tweet* [65] can fill this roll. The primary challenge remains dissemination among the large numbers of people concentrated within the towns or cities close to the events.

Addressing: The intended audience for censor-triggering content is usually the general public, so traditional addressing schemes are an unnecessary complication. For example, in Twitter, tweets are addressed to followers [4]. In Shout, messages are delivered to whomever is nearby, simply and efficiently reaching the (much broader) target audience.

Content: The content is intended for public dissemination, so support for message confidentiality[5] is unneeded. All messages can be broadcast in plaintext.

Routing: Routing decisions are pushed onto the humans using Shout. Although messages are intentionally broadcast to the general audience, they should still be restricted to portions of the network with a high density of interested users. For example, content of interest only to people in one neighborhood should not flood others. Complete, automatic identification of regions of interest for particular messages is not yet feasible, so Shout uses human involvement instead. Messages will naturally spread within regions where the reshout rate is high—the content presumably interesting— and will die out in regions where it is not. Automated reshout techniques can amplify the reshout rate to speed dissemination.

Adoption: Shout attempts to provide value to users not concerned with censorship to increase the likelihood of widespread adoption. Rapid adoption of a new application by the general public immediately after an increase in censorship is unrealistic. By delivering messages based on geographic proximity, not social relationships, we hope Shout will be useful in everyday life as well. For example, a shout mentioning leftover food in a conference room would be implicitly sent to those near enough to get the food, as opposed to an email to a listserv that includes people currently out of the building.

[4] Anyone may browse a user's stream, unless set to private, but the intended primary delivery mechanism is through the *follower* relationships.

[5] Confidentiality differs from sender authenticity or sender anonymity, which we discuss later.

3.2 Overview

Shout is designed around the premise that for censorship-resistant microblogging, the communication style can adapt to the most natural network architecture. In particular, a broadcast protocol is appropriate, unlike traditional systems that address messages to particular users or groups. Second, the participants are motivated to share content, so a user-controlled mechanism for spreading messages, similar in principle to gossip in the real world, is appropriate.

Ad hoc networks provide good resistance to censorship, but suffer from reduced throughput, increased latency, and poorer routing scalability. The non-hierarchical structure implies that censoring communication would require controlling many of the participating nodes, a task much too expensive when the devices are the smartphones already carried by many people. But it also hurts the traditional network performance metrics. As networks grow, more traffic must flow through devices with constrained bandwidth and each message is routed through more devices [38]. Fortunately, for microblogging the communication structure can be adapted to the ad hoc structure, mitigating the impact of those scalability concerns.

We have the following goals for Shout.

Unblockable: A centralized authority should not be able to selectively block most users from sharing messages without also blocking significant, legitimate traffic. In essence, we wish to prevent technological means for censorship. We do not explicitly attempt to prevent self-censorship due to fear of reprisal, but describe in Subsection 3.4.2 how the non-hierarchical structure does provide some advantage here too.

Efficient: Total network traffic should scale with message reach. The limited throughput available in the ad hoc network should be concentrated on messages of widespread interest or importance.

Verifiable: As an open system, anyone can send messages, good or bad, true or false. Thus, readers must be able to verify the authorship of each message. Note that verifying the real-world identity of an author is not necessary. Often, simply ensuring that a message

came from the *same* anonymous individual who published true, useful information in the past is sufficient to have reasonable trust in its content.

Adoptable: Systems useful only for sharing censorable content or only during times of extreme Internet blocking are not widely used by the general public. Providing features useful in day-to-day life and not seen in existing applications will help with adoption.

Simple and Extendable: True solutions evolve from earlier efforts, and we hope Shout is just one step in such a chain. Thus, it must be simple and extendable, allowing others to invent, implement, and test future innovations and improvements.

3.2.1 Threat Model

We consider the following threat model with respect to our goal of unblockability. The censor is assumed to be a state authority with control over infrastructure, e.g., top-level Internet routers within the country, wanting to limit the spread of certain information, e.g. that critical of the government. With control of the infrastructure networks, the censor can employ any of the numerous techniques developed to detect and block potentially-objectionable content.

We assume the government allows the widespread use of WiFI-equipped computers and smartphones, perhaps due to their significant economic benefit. WiFi transmissions can be blocked by jamming, but we assume the censor cannot jam large portions of the network. Doing so requires covering large geographic regions with jamming transmitters and thus is quite expensive. Further, it seriously disrupts legitimate uses of the airwaves. Selective jamming could block just the questionable traffic, but require specialized jamming equipment, further increasing the cost.

Finally, we assume the censors do not control or mandate the installation of special software on the devices running Shout, e.g., the laptops and smartphones. Although theoretically possible, recent failed attempts, like China's Green Dam Youth Escort initiative [66, 67], highlight the practical difficulty.

3.2.2 Applications

The primary motivation for Shout is censorship-resistance, but the design supports a variety of applications in which the independence from infrastructure or geographically-based delivery is beneficial. Such applications may help encourage early adoption of Shout, such that it is already in use and available when its censorship-resistance is needed. We briefly mention a few here.

As stated before, Shout is useful for censorship-resistant microblogging. The non-hierarchical structure is difficult to shutdown or block, as Twitter was in Egypt in early 2011 [14]. Further, messages do not flow through intermediate choke-points, eliminating opportunities to selectively censor certain posts.

Shout is also useful whenever infrastructure is unavailable. For example, after a natural disaster that destroys cellular towers or knocked out power, the ad hoc system would continue to function. Authorities could broadcast safety instructions. Victims could self-report their locations and condition to others nearby. First responders could read incoming Shouts to assess the situation without waiting to interview bystanders.

Similarly, consider dissemination of public safety messages about ongoing gas leaks or tornado warnings. Text-messaging is commonly used for this purpose, but people in many buildings get poor, if any, cellular coverage. The University of Michigan Emergency Management Team is interested in applications like Shout, that extend the reach of their emergency broadcast system into basements and laboratories with no cellular coverage [68]. Despite a very different goal than censorship-resistance, the same architecture is an attractive solution to this problem.

Finally, Shout has potential for day-to-day use as well[6]. Messages flows via a geographic network—broadcast to other nearby users—instead of via social links. This makes Shout ideal for an ephemeral audience determined by proximity. For example, concertgoers, sports

[6]Many of these ideas could be accomplished with location-aware Internet-based services as well. An ad hoc network, though, supports them naturally and without the privacy concerns.

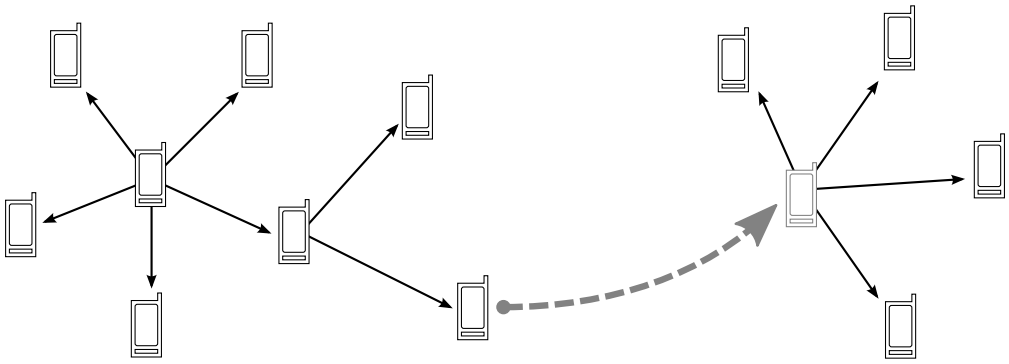


Figure 3.1: Shouts are broadcast to one-hop neighbors. A recipient interested in the message can reshout, or rebroadcast, increasing the effective range. Additionally, one can reshout after moving to a new location, reaching otherwise-isolated portions of the network. Automatic rebroadcasts ca increase the dissemination rate.

fans, or conference attendees—people at a common event—would easily and naturally see each others tweets during the event. No social relationships are required and none persist after the event. Similar behaviors have emerged with Twitter, with groups choosing a specific hashtag so tweets from the event are easily searchable. With Shout, even that effort is not necessary.

Students might use Shout in various ways. For example, consider one struggling with a general chemistry problem set in the library. He can send a message asking for help to locate other nearby classmates willing to help—classmates that, in a class of hundreds, he probably doesn't already know. Or consider a pickup game of football that needs a few more players. It's easy to broadcast that request to nearby people, strangers included, that might want to join.

3.2.3 Design Summary

Shout is designed for smartphones that can communicate via ad hoc WiFi. When a message or *shout* is sent, it is broadcast to the one-hop neighborhood, as illustrated in Figure 3.1. To avoid wasting limited bandwidth and energy on uninteresting content, messages are not automatically transmitted further. Instead, much like gossip in the real world, a recipient can

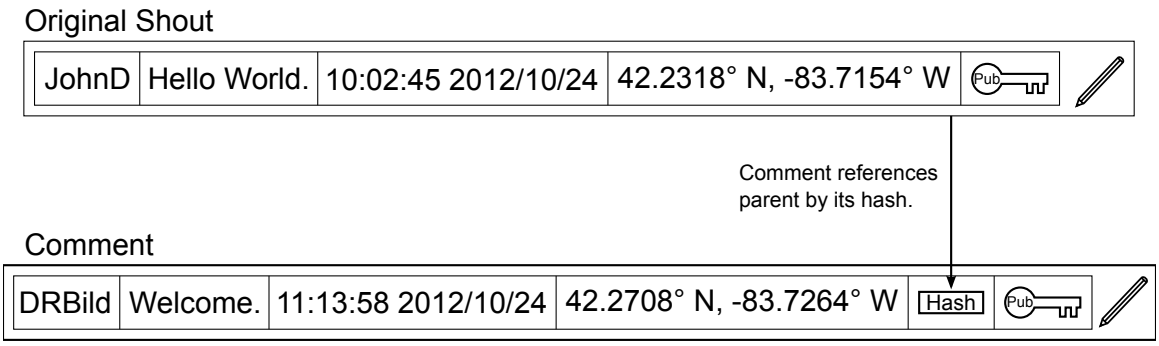


Figure 3.2: Each shout contains a user name, message, timestamp, location tag (optional), the sender's public key, and a self-signature. A shout intended as a comment on a prior shout references that parent via a hash of the parent.

manually *reshout* a message to increase it's range to his one-hop neighborhood. Reshouting after moving to a new location can extend the reach to otherwise disconnected portions of the network. To ensure widespread distribution, someone could act like a town crier, intentionally moving from place to place, reshouting in each. Using manual intervention for further broadcasts helps ensure that content only propagates through the portions of the network with interested users. Within these regions, automatic rebroadcasts can be used to reduce delivery latency.

Figure 3.2 illustrates the common information included in a shout. As with traditional services, each contains a username, message, timestamp, and an optional location. Unlike traditional services, these fields are set by the sender and thus could be falsified. As a decentralized system, usernames are not unique, so the contents are self-signed with an included public key. This public key serves as an unforgeable identifier, so one can determine whether two messages claiming the same username actually came from the same source. Finally, a shout may reference a prior shout by hash. For example, comments include a hash of the parent shout.

Shout is a fully-decentralized system, so information is local to each device, i.e., a user's smartphone, as illustrated in Figure 3.3. In particular, no global database of past shouts is maintained. Each device stores the shouts it has heard, but because users have

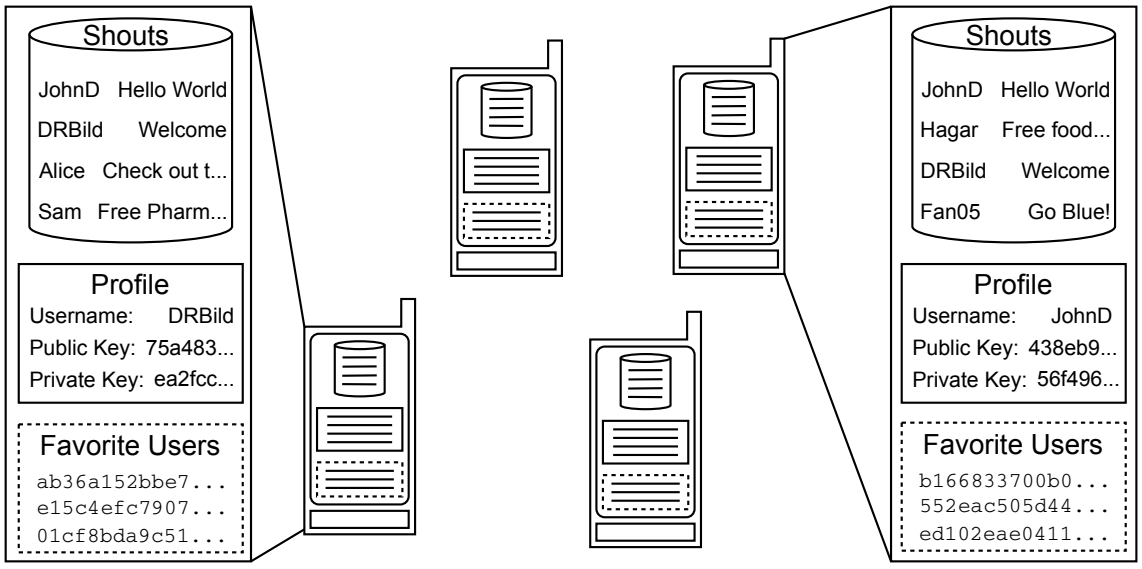


Figure 3.3: Shout is fully-decentralized so information like past shouts and one's user profile is local to each device. Only shouts one has heard are available, so each device has a different partial view of the history. Features like lists of favorite users must also be managed locally.

different location histories, most will have observed different sets of shouts. Consequently, any analysis or "view" of the world derived from the database can vary from user to user. Features usually performed by a central server, like spam filtering, search, or authorship verification must instead be performed locally.

3.3 Decentralized and Non-Hierarchical Architecture

In this section, we describe the Shout architecture and protocols, paying particular attention to why these design decisions were made. First, we justify the decision to base Shout on ad hoc WiFi networks and describe how this informs later design decisions. Subsequently, we discuss our solution to decentralized identity management. Then, we describe the details of the Shout network protocols, both for sharing messages and larger content like pictures. Finally, we briefly discuss local message management, i.e., search and filtering.

3.3.1 Ad Hoc WiFi

Decentralized microblogging services have been previously proposed [69–71], designed to improve reliability and increase scalability by reducing the dependence on a centralized provider. Unfortunately, these solutions are insufficient to address our primary concern—censorship—because they still rely on a hierarchical delivery mechanism, the Internet. In particular, these solutions assume that communication costs are similar between all pairs of users. Hierarchical structures approximate this property, but non-hierarchical networks, in which transmissions between distance nodes must pass through all the intermediate nodes, do not [38].

Hierarchical networks are inherently susceptible to censorship, because much traffic flows through a few centralized points at the highest levels. These chokepoints are a prime location to efficiently effect censorship and surveillance. Instead, we use non-hierarchical networks, for which similar behavior would require controlling many of the participating devices or communication links. We believe such control is too expensive or economically damaging to be of concern.

We based Shout on ad hoc WiFi, a non-hierarchical networking technology already widely deployed. The prevalence of existing hardware support means that Shout is easily deployed as a software installation[7], significantly increasing the chances of real adoption. In short, we chose ad hoc WiFi for its censorship-resistant, non-hierarchical structure and its existing availability.

The choice of ad hoc WiFi influences many aspects of the design. The range of a single transmission is short, 50–100 m and multi-hop throughput does not scale [38], so most communication must be local. Generally, transmissions should be of interest to the recipients, not just intermediate hops on the path to an interested receiver. Individual

[7]Some platforms, like Android, disable the ad hoc mode in software. These limitations are easily removed by a software update and can be worked around by rooting the phone. We hope that apparent support for the emerging WiFi Direct standard [72] signals that manufactures will better support ad hoc connectivity in the future.

transmission sizes are limited, usually to 1500 bytes[8]. Typical Shout broadcasts should fit in such packets. All transmissions are effectively broadcast, so for highest efficiency, all messages are public and readable by any device in range of the transmitter. We do not naïvely support encrypted messages in Shout. Finally, routing schemes do not scale with network size, as routing table maintenance consumes an increasing fraction of network bandwidth [40, 41]. Thus, we do not support addressing of messages—like friends and followers in social networks—in Shout[9]. Messages are assumed to be intended for those nearby—content of broader interest can propagate further via reshouting.

3.3.2 Identity Management

Identity management in a decentralized system is not trivial. In a service like Twitter, one trusts the centralized system to ensure that usernames are unique and only the true owner of an account can post. Without such an omnipotent authority, the desired properties must be explicitly enumerated and incorporated into the protocol.

The first task is determining the purpose and features desired for identities in Shout. At a high level, we wish to support the notion of authorship. Each message should be associated with an authoring entity, so that messages from the same entity are easily grouped and those from different entities easily separated. Further, the authorship should be verifiable. Such verification is useful in two ways. First, it allows confirmation that a message purporting to be from a particular person, say a friend, is not a forgery. Second, it allows the development of anonymous entities known only by their posts. For example, confirming that a message containing surprising, hard-to-believe information came from an otherwise-unknown entity who has only posted true things in the past might increase one's belief in the new message.

Thus, we desire an identification scheme that is decentralized (i.e., no central authority

[8]WiFi supports larger MTUs, but the the Ethernet MTU of 1500 bytes is usually used, on the assumption that transmissions are Internet-bound and thus eventually traverse an Ethernet link.

[9]Shout could be easily extended to support tags on messages—much like hashtags in Twitter—-that could be used for content-based addressing. We don't believe this is critical for censorship-resistance and thus is left as future work.

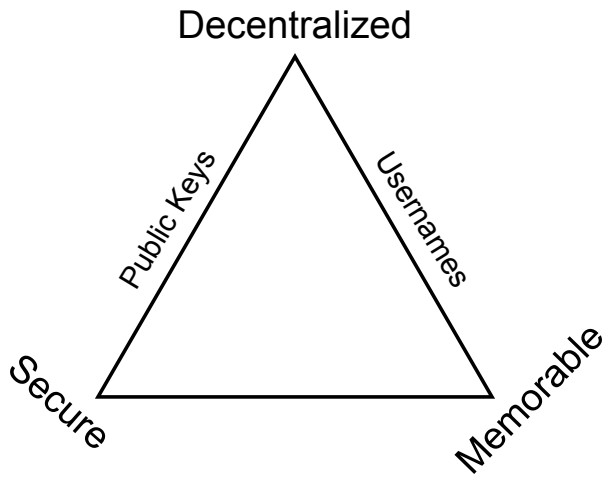


Figure 3.4: Zooko's triangle [2]. A single naming scheme can include only two of the properties. The Shout protocol uses both self-chosen usernames and public keys to incorporate all three properties. Third identifiers can be generated locally to provide unique names that are easy for humans to compare and remember.

is needed to issue them), secure (i.e., authorship is not forgeable), and memorable (i.e., humans can easily remember and identity important names). Further, the scheme should be simple and not require significant network resources. Unfortunately, a scheme with all three properties is not believed possible.

These properties are known as Zooko's triangle, illustrated in Figure 3.4, and the general belief is that a single naming scheme can have only two of the properties [2, 73]. Thus, for Shout, we employ two naming schemes, self-chosen usernames and public keys. In typical situations, with most network participants behaving, the usernames, which are decentralized and meaningful, will be sufficient. Should two nearby people pick the same name, one will likely change to reduce confusion. Only in the case of intentional impersonation is such duplication a serious concern. For this, Shout employ public keys, which are decentralized and secure. Each message includes both a username and public key and is signed by the corresponding by private key. These signatures serve to prove the authorship of a message. Users wishing to verify authorship against real-world identities can exchange public keys out-of-band, much like PGP.

Of course, public keys and signatures are not intended to be human-readable. Without

help, most users will likely rely on the username only, missing possible forgeries. We summarize several possible solutions next.

The program could proactively warn users about duplicate usernames. When a displaying a shout with a username used by multiple public keys (in the local set of shouts), a warning could be display to the user. The user could then compare the shouts sent under each of the public keys to help determine the actual author identity. This sort of solution places significant burden on the user and is likely to be disabled or ignored.

Instead, displaying a name that spans the other leg of Zooko's triangle—one that's secure and meaningful—is best. These cannot be global and thus must be local to each user's device. For example, consider using color as the identifier. A different color could be locally assigned to each duplicate user and displayed as a border or background on the shout. Textual names are perhaps more memorable. Steigler proposed a system along these lines with his Petname system [73]. Here, one picks a local identifier, or petname, to correspond to the global identifier, or public key, and the system translates between the two. That is, the local petname is displayed instead of the public key. As long as the user assigns distinct petnames, they are secure and meaningful.

Regardless, the Shout protocols are independent of these solutions. Shout supports the two legs of the triangle possible in a decentralized system and is easily extended to incorporate local systems for the third. Existing key-exchange and signature verification programs, like web-of-trust and PGP, are compatible with Shout. Further, arbitrary third-party solutions for mapping public key to local secure and memorable identifiers can be used. We hope Shout serves as a platform to test various solutions to this problem with real users.

Some have authors have tried to "square the triangle", by proposing identity schemes that claim to have all three properties [74]. For example, Namecoin [75], a distributed DNS-alternative based on Bitcoin [76], uses hashchain-based proof of work to generate

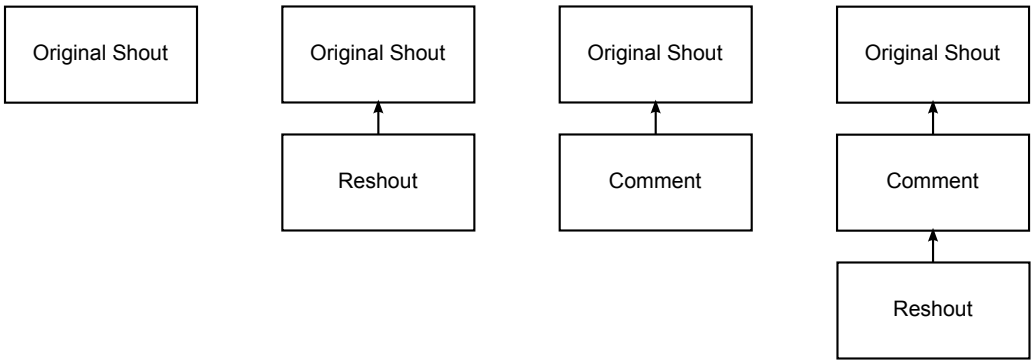


Figure 3.5: The three types of shouts and their relationships. Comments are restricted to a single level so that the largest full chain (a reshout of a comment) will fit in one WiFi frame.

globally unique and secure mappings between public keys, URLs, and addresses[10]. These schemes require that changes to the global hashchain be propagated to all users, and thus are not suitable for limited-throughput ad hoc networks.

3.3.3 Messages

Two primary considerations directed the design of the Shout message format. First, each transmission should be less than 1500 bytes, to fit the MTU of real-world WiFi devices. Second, each transmission should carry the full context for the message, e.g., the prior shout if the message is a comment. These requirements tightly constrain the information that can be fit into a shout and the length of comment chains.

Figure 3.5 show the three types of shouts. *Original* shouts are stand-alone, new posts. A *comment* is a new message that also references an original shout. When the comment is broadcast, the original shout is included in the same transmission so the conversation context is guaranteed to be available to the recipient. Comments may not reference another comment, because the context chain would be too long. A *reshout* is more than just a rebroadcast of an existing shout. It contains all information of regular shout except a message, but references the original shout (or comment) being reshouted. Again, the full chain is transmitted.

[10]Under certain assumptions about the relative computational power of attackers to conforming participants.

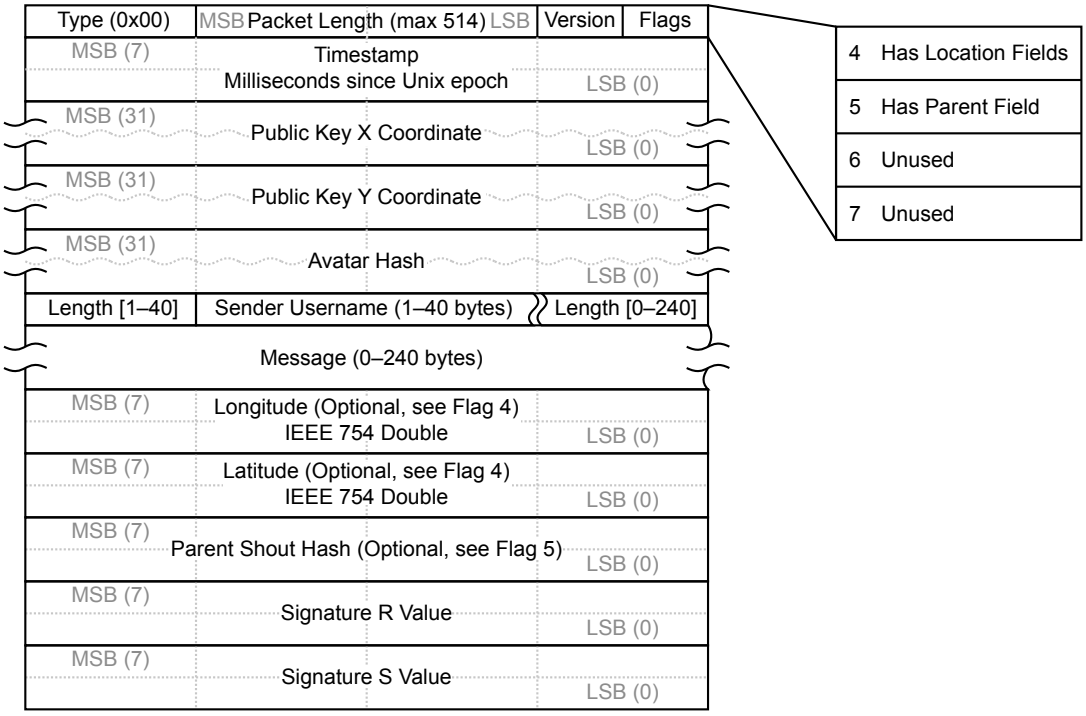


Figure 3.6: The network packet format for a shout. The hash used to reference a shout is also computed over this canonical form.

Each shout contains the fields one would expect for a microblogging application, as show in Figure 3.6. The user is identified by a self-chosen username and public key, as discussed in the preceding section. Avatar images are too large to fit in a packet and are unlikely to change frequently, so only a hash-based reference is included. Subsection 3.3.4 describes the protocol for retrieving the actual image. A timestamp indicating the time of sending is included, although recipients have no way to verify this time. The location from which the shout was sent may be included, but again, it cannot be verified. The message contents are limited to 240 bytes, to fit the 1500 byte limit. If a reshout or comment, the parent is referenced by including its SHA-256 hash, taken over the canonical network format of the parent. Finally, the contents are self-signed. This signature can be verified using the public key field.

Shout uses elliptic curve cryptography for digital signatures, because keys and signatures are shorter than for RSA. A fixed curve, *secp256r1*, is used so to save space—the curve

name does not need to be transmitted. This makes changing the curve or signature algorithm in the future more difficult, but given the nature of our application and the forecasted lifetime of 256-bit ECC [77], we think the tradeoff is reasonable. The public keys are included uncompressed—both the x and y coordinates are given in full.

3.3.4 Content Sharing

Internet-based microblogging services support user avatars and the referencing of additional content via hyperlink in the message body. References to pictures, in particular, are often automatically dereferenced, the image displayed inline with the message. Due to their ubiquity in the online world, we believe these features are necessary in Shout to help adoption, but implementing them in an ad hoc network is much more involved. As already mentioned, bandwidth is limited, so transmitting kilobytes or megabytes of image or content with each reshout is infeasible.

To reduce the bandwidth demands, images[11] are instead shared asynchronously and on-demand. Avatars change infrequently—most users will send many shouts with the same avatar. Thus, a particular avatar need propagate through the network only once. On subsequent shouts, it is already available locally and need not be re-transmitted. Attached images may not see the same reuse, but the asynchronous, on-demand sharing still ensures that the content is transmitted only when a recipient first requires it.

To ensure integrity, images are referenced in shouts by a cryptographically-secure SHA-256 hash. Thus, the digital signature of the shout covers the image as well. The avatar hash has its own field in the shout message format. Other images are included as URIs in the message body, much like hyperlinks in Twitter, in the form `shout://<hash>`, where `<hash>` is the 64 character hexadecimal encoding of the content hash.

Were the hash reference taken directly over the image, one would have to receive the

[11] The described scheme can share arbitrary content, but we envision images as the most popular use. We use image in the remainder to ease explanation.

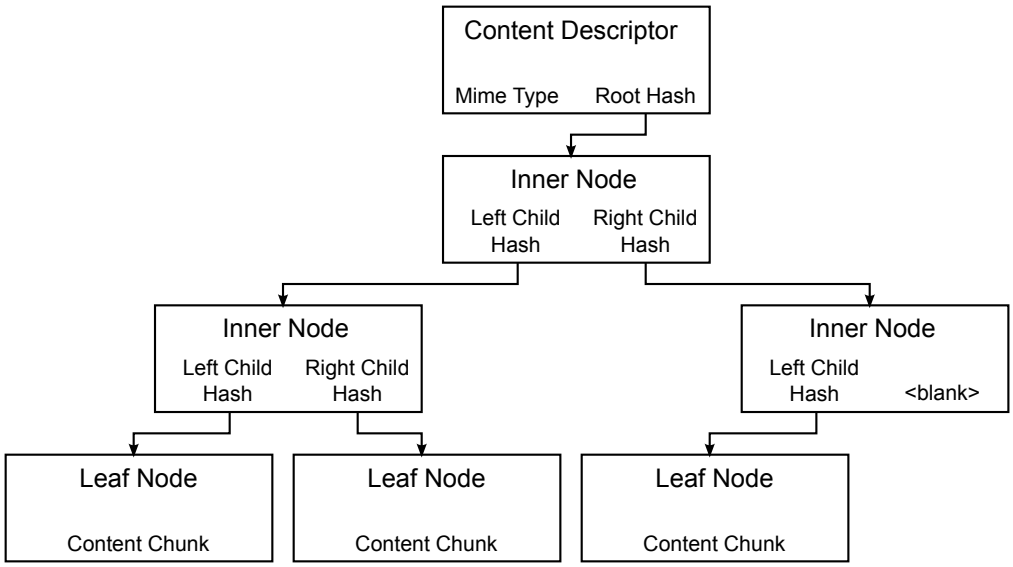


Figure 3.7: Hash tree mechanism used to reference and distribute images and other large content in Shout. The leaf nodes are packed to the left and contain the content is sequential order. The content descriptor includes a MIME type, so that hash references to the tree specify both the content bit string and how it should be interpreted.

entire content to verify its correctness. This opens a possible attack. In response to a request for the image referenced by a given hash, an attacker could respond with an arbitrarily large amount of incorrect content and the receiver would be forced to store it all, unable to check its correctness until all was received. Instead, the hashing scheme should allow the correctness to be verified at each transmission, so that falsified chunks can immediately discarded.

Shout uses a hash tree to obtain this property, as illustrated in Figure 3.7. The content is split into chunks of no more than 1450 bytes each—with headers, this fills the 1500 byte MTU. These form the leaves of a binary tree, with each parent node containing the hashes of its two children. The veracity of a particular node can be checked given only the hash contained in its parent—no other portions of the tree are necessary. The packet formats for leaf and inner nodes are shown in Figure 3.9. Hashes are taken over the entire packet contents, as shown in Figure 3.8.

It is important that the recipient interpret the received binary content in the intended way.

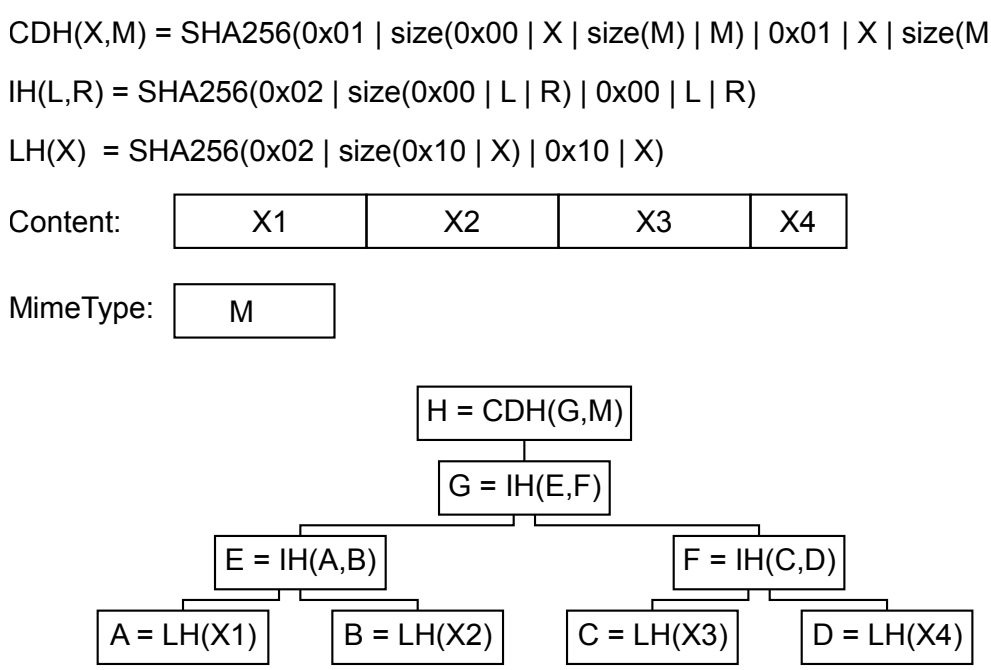


Figure 3.8: Example hash tree for content four data blocks long ($X1$, $X2$, $X3$, and $X4$) and with MIME type M. The hash H would be included in the avatar field or Shout URI. The SHA-256 hashes, computed over the canonical network format shown in Figure 3.9, are defined here for clarity.

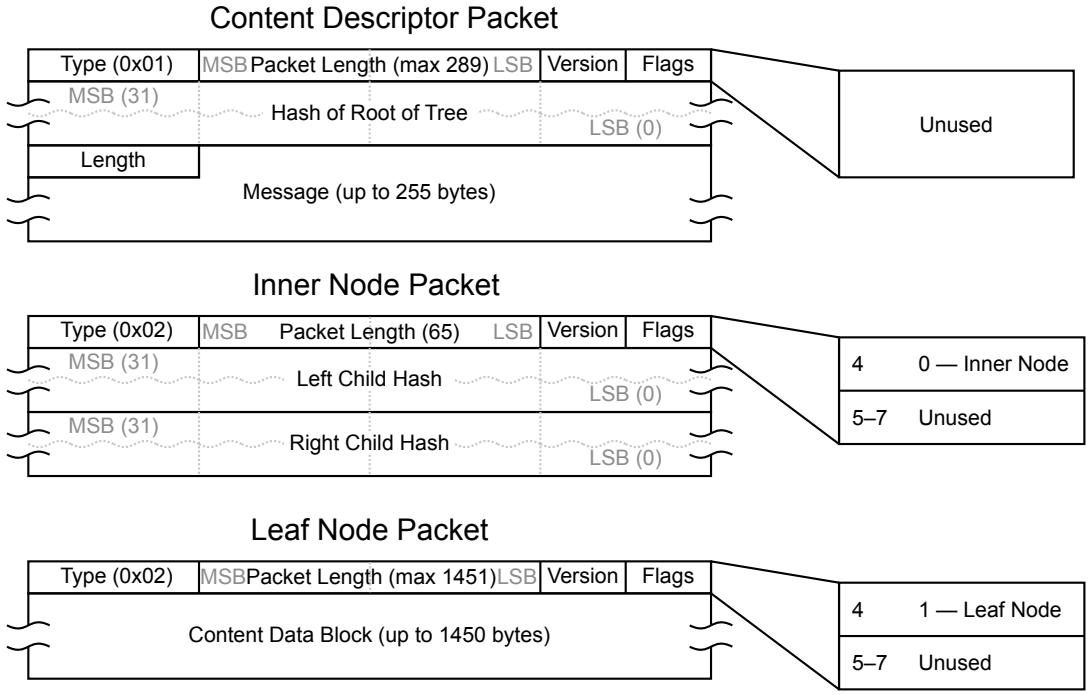


Figure 3.9: The network packet formats for content descriptors and hash tree nodes.

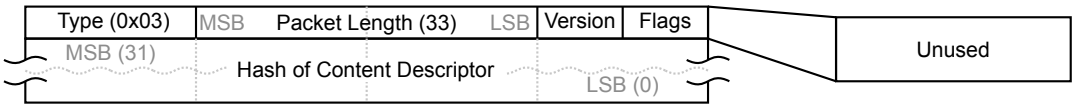


Figure 3.10: The network packet format for content requests.

A bit string might be, for example, both a valid image file and a valid, but malicious, executable. A trusted sender might reference the image and a recipient tricked into interpreting it as an executable. To prevent this, Shout embeds a MIME type into the hash tree for each piece of content, so that the digital signature of the shout covers not only the content, but also how it should be interpreted. This *content descriptor* is illustrated in Figure 3.7 and Figure 3.9. It contains both the MIME type of the content and the hash of the root of the tree. The avatar field and Shout URIs reference the hash of the content descriptor.

The content descriptor and hash tree packets are transmitted on demand. When a client tries to view an image it does not have, Shout sends a content request packet, shown in Figure 3.10, to request it. Any one-hop neighbors with that content will respond. Responses are randomly delayed to reduce collisions and if a valid response from another device is overhead, the response is not sent. If no neighbor responds, the request is retried with exponential back-off.

On the assumption that a device missing a parent node is also missing its children, the subtree rooted at the requested node is sent proactively. For the typical case, where the entire tree is needed, this requires only a single content request packet, instead of one for each tree node.

The content is most easily available when the shout referencing it is first heard—the node sending the shout likely has the content—so it is proactively requested then. Each incoming shout is scanned for avatar and image references and those that are unavailable (or partially unavailable—some tree nodes are missing) are requested immediately.

This system does not guarantee the availability of avatars, images, and other content, because content is requested only from one-hop neighbors. Although this request range

could be extended at the cost of additional bandwidth and energy, we think the one-hop neighborhood provides a good tradeoff between efficiency and availability.

3.3.5 Message Management and Filtering

Internet-based microblogging services employ a variety of means to help users sift through the flood of posts for the ones they are interested in. The two most common are user whitelisting (*following* in Twitter parlance) and search. All posts from whitelisted or followed users show in one's main feed, providing an easy way to specify the exact sources to listen to. Search provides an easy way to find recent posts about specific events or ideas. Hashtags, user-directed labels, facilitate such searches. Further, the services filter spam and fraudulent posts to increase message quality. In Shout, all such filtering must be done locally.

All overheard shouts are included in the local database, so searching and filtering is largely independent of the Shout protocols. The local application responsible for displaying shouts to the user can support arbitrary methods independent of other Shout users. This makes Shout an excellent platform for experimenting with search and filtering ideas.

We believe that persistent, user-defined search queries can fulfill the same features offered by Internet-based services. For example, a search query that selects all shouts from a group of users is essentially equivalent to a follower-based Twitter feed[12]. Similarly, persistent searches for particular keyword, hashtags, or locations offer alternative methods of subscribing to certain shouts.

Spam is a more vexing problem. Although the follower-like search queries mentioned previously offer a method to whitelist certain senders, effectively hiding spam, they hurt one of the primary motivators for Shout. Shout should be useful for exchanging ideas with nearby strangers—people one has no knowledge of or reason to whitelist. Thus, a different

[12]The primary difference is that such a search can return only the shouts heard, not necessarily all shouts sent by that group of users

approach for spam filtering is needed.

Chapter 6 develops a spam detection technique, but we briefly describe the intuition here. Spam filtering can be done in two ways, blocking content (spam) or senders (spammers). We believe the first is too difficult to do automatically, as messages are short and spammers clever [78]. Shout offers natural resistance to the spread of spam—most people will not reshout junk and thus it will not spread—but it will still annoy people in the one-hop range of the spammer. Thus, we focus instead on identifying and blocking spammers.

Blocking of spammers comes in two fashions, whitelisting and blacklisting. With blacklisting, all senders are presumed innocent and only blocked after exhibiting behaviors of a spammer. With whitelisting, all users are presumed guilty and only unblocked after exhibiting behaviors of a non-spammer. Blacklisting is useless in Shout, because the spammer can simply create a new identity once blacklisted[13]. Thus, we are forced to consider whitelisting.

We have already ruled out explicit manual whitelisting, because strangers will not be whitelisted. Instead, we develop an implicit whitelisting strategy based on reshouts. Intuitively, non-spammers should be reshouted more frequently and by more users than spammers. Consider a graph with users as nodes and a directed edges representing that one user reshouted another. Non-spammers should be more-connected in this graph and have shorter paths between them. Our strategy classifies spammers and non-spammers according to their connectivity in this graph.

Spammers are free to create arbitrary connections between their own identities, altering that portion of the graph. To combat this, the graph is rooted at the user doing the spam filtering. The spammer identities, no matter how connected amongst themselves, should still have low connectivity to this trusted node.

This approach requires some bootstrapping. The graph can be constructed only over the locally-available shouts. When first joining Shout, this set is small. Instead, one can prime

[13]The Shout software can still support blacklisting, as it may be helpful against some advertisers or otherwise annoying users. But it is not, on its own, a sufficient defense against spam.

the set by retrieving all shouts from a trusted friend or acquaintance who has been using Shout longer.

Further, a new user joining Shout will not have been reshouted and thus will not be connected in the reshout network. We deal with this in two ways. First, friends can manually whitelist the new user. They will see his shouts and, if appropriate, reshout, building the users reshout connectivity. Second, some users may wish to browse the unfiltered timeline of shouts and, upon seeing good content, reshout it. We suspect that if the spam filtering strategies are good, the unfiltered timeline will still be relatively spam free. (Remember that spammers can only reach one hop, so spamming many locations is at best complex and at worst very expensive).

3.4 Security Analysis

This section describes several attacks on Shout and its primary goal, censorship-resistance. Censors can employ two classes of techniques against Shout. First, they could block transmissions by technical or legal means. Second, they could fine, imprison, or otherwise harm individuals using the system such that a fear of reprisal discourages further use. We discuss both classes and describe how Shout defends against or mitigates these attacks. Some of these attacks are outside of our attack model—largely because we believe them infeasible—but are mentioned here for completeness.

3.4.1 Censorship by Blocking

The most obvious technical means to block Shout transmissions is to jamming the radio signals. Although technically feasible, Shout's distributed nature mitigates this risk. Blocking most Shout traffic require jamming the airwaves around most users, a very expensive and disruptive proposition. Legitimate and economically-important business uses of WiFi would also suffer. Practical jamming attacks will be limited to small regions and thus not a serious

concern for our distributed architecture.

To avoid blocking allowed traffic, censors could instead employ selective jamming, blocking only those transmissions that appear to be Shout content. This method has even greater expense—the airwaves around most users must be monitored and jammed using more-sophisticated, and thus more-expensive, jamming equipment—so again we do not find this to be a serious practical concern. Steganographic techniques for Shout traffic to masquerade as other legitimate traffic (e.g., standard encrypted AP-based WiFi traffic) could be developed, but we believe the costs of the increased complexity outweigh the near term risks of selective jamming.

A potentially more cost-effective approach is to mandate that smartphones (laptops, etc.) come equipped with software that blocks the installation or use of tools like Shout. Again, although technically feasible, we believe this is practically difficult. This type of cenosrship is much more publicly visible than filters on top-tier routers and requires the direct cooperation of many people in the supply chain. Public response and disagreement is much more likely. History supports our view. The Chinese government mandated that by July 2009 that every computer sold in China must include the Green Dam Youth Escort software content filter [66, 67]. The law prompted significant criticism levied at both the moral implications of such a requirement and practical flaws in the software itself. Shortly before it was to take effect, the mandate was postponed and, as of 2013, has not been reinstated.

As an extreme approach, a government could ban the sale of devices containing WiFi transceivers. Again we think this is infeasible in the long run, as wireless network access is both extremely popular among the public and important to many businesses. China has had some success in mandating that WiFi devices sold there support the government-approved WLAN Authentication and Privacy Infrastructure (WAPI) protocols, custom alternatives to the standard 802.11b and 802.11i security protocols [79]. Some devices were initially sold without WiFi capability, but demand has lead to later models including it. Use of WAPI,

instead of 802.11i, appears scarce.

3.4.2 Censorship by Reprisal

A more concerning avenue for censoring Shout users is reprisal—people concerned for their property, safety, or freedom are more likely to self-censor. Although Shout is not designed to completely eliminate such concerns or provide a strong notion of anonymity to senders, we describe here the extent to which such anonymity is possible in Shout.

Perhaps the most important protection is that users cannot be required to explicitly link their real-world identities to those in Shout. China is attempting to force users to register with Internet services using their real names [80] and Saudi Arabia is considering similar legislation for Twitter[14] [81]. In Shout, identities (usernames and public keys) are decentralized and changed at will; no authority can mandate any structure or content. That Shout posts do not (have to) contain directly identifying information significantly increases the challenge of identifying those posting messages.

Senders can still be identified by the location of transmissions, albeit at much greater difficulty and expense. Triangulation methods can identity the precise location of a transmitter. Simply observing a shout in some location reduces the anonymity set of the sender to those within WiFi range, 50–100 m. More sophisticated traffic analysis can reduce the anonymity sets further. For example, one could correlate the multiple locations of objectionable shouts to find the people who frequent both locations—say home and work.

Although Shout does not directly protect against these schemes, it does reduce the risk. Monitoring and collecting all traffic is prohibitively expensive, so such attacks are likely to be targeted at specific individuals, not levied against the entire population by large-scale, preemptive data analysis. For users already on government watchlists, Shout may be too risky. For typical people, it should offer a method to communicate free of the censorship often imposed on Internet-based services.

[14] How this would be implemented or enforced is unclear.

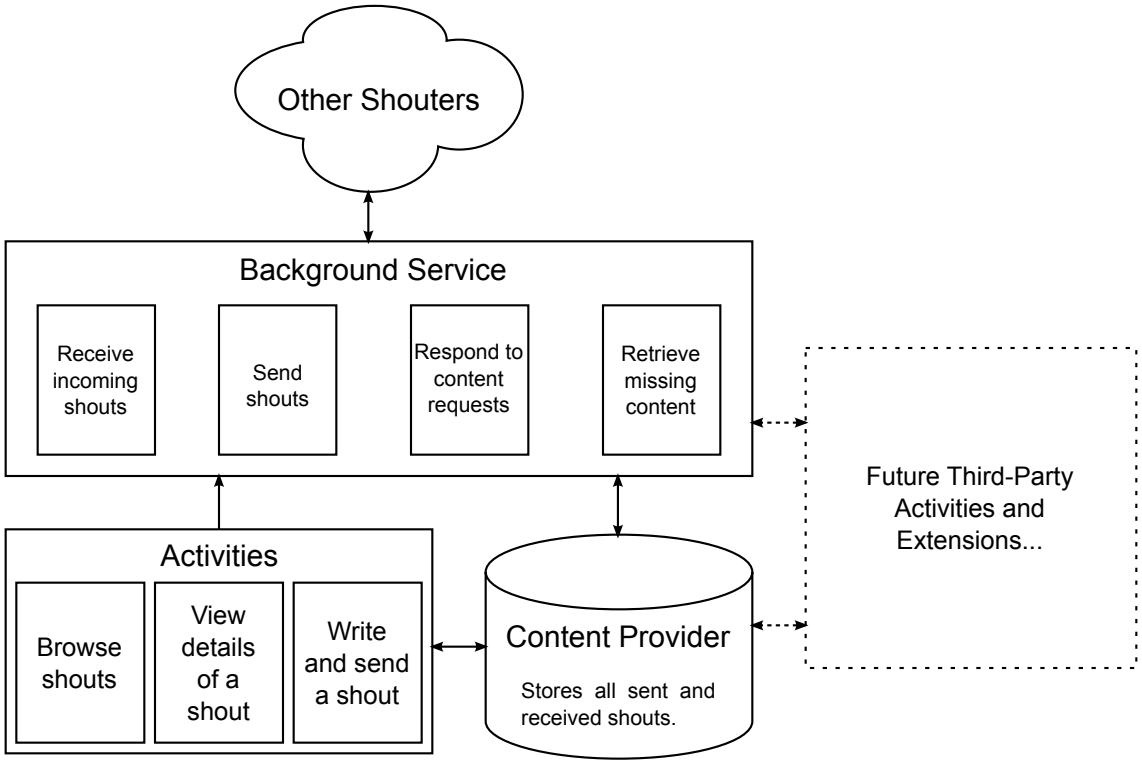


Figure 3.11: Architecture of Shout implementation for Android.

3.5 Implementation

This section describes our implementation[15] of Shout for Android smartphones. The Shout protocol could be implemented for other platforms as well, e.g., iOS. We only require support for device-to-device ad hoc communication. At this point in time, it is not clear which mobile platforms will provide the best support moving forward. We chose Android for our prototype because it is the most popular smartphone operating system [82], has good support for background services and extensible applications, and is open-source, potentially useful for future experimentation or research.

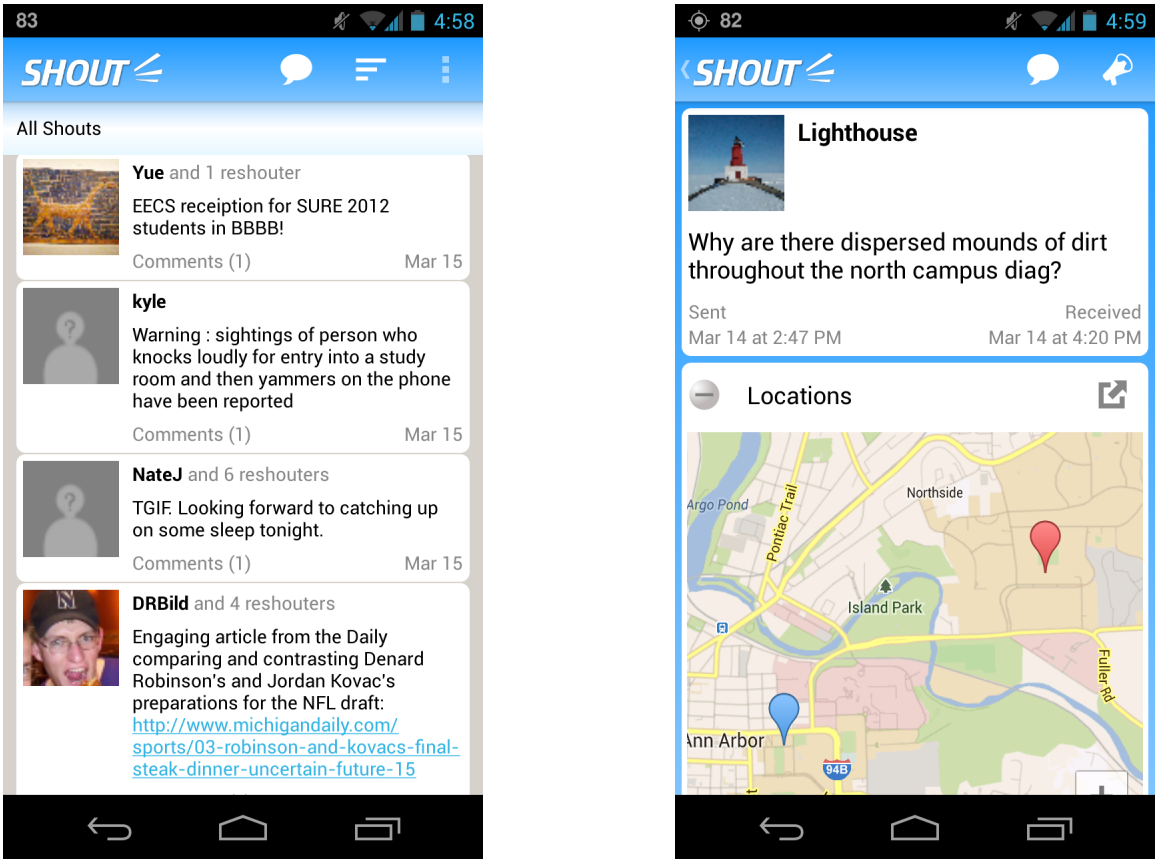


Figure 3.12: Screenshots of the Shout activities for browsing received shouts and viewing detailed information about a specific shout.

3.5.1 Implementation for Android

Figure 3.11 shows the application architecture. The activities are the main interface and display for users, the service runs in the background listening for new shouts and responding to content requests, and the content provider stores and provides access to the received shouts.

This architecture is intended to be extensible. The service behavior and provider contents are largely determined by the Shout protocols, but other activities may interact with them. We hope others experiment with, improve, extend, and maybe even replace the activity components.

[15]Many people contributed to the implementation, including David Adrian, Nate Jones, Yue Liu, Gulshan Singh, Anthony Tesija, Jonathan Tiao, and Bowen Xu

Activities: Screenshots of the two main activities are shown in Figure 3.12. The timeline activity shows an ordered list of received shouts. Tapping on a shout reveals any comments and buttons for reshouting, adding a comment, or opening the details view. The details activity shows extended information about a shout, including when it was received, a map of its location of those comments and reshouts, and a list of the reshouts.

Service: The service runs in the background listening for new shouts. When new shouts arrive, they are stored in the content provider. The service manages the exponential backoff policy for requesting missing content and responds to content requests, if that content is available.

Content Provider: The content provider stores and provides access to the shouts. Other applications may access the provider, so methods of displaying, filtering, or analyzing the shouts not directly supported by our release are easily added.

3.5.2 Practical Implementation Concerns for Ad Hoc WiFi

Unfortunately, Google has disabled the ad hoc feature of WiFi, so it cannot be used without rooting the phone. Instead, Shout is deployed on MANES (Chapter 4), a mobile ad hoc network emulation system. MANES estimates the ad hoc topology of client devices by monitoring their locations and visible WiFi access points. Packets intended to be broadcast over the ad hoc WiFi are instead send to the MANES server, which relays them to the devices estimated to be within range. Shout itself does not depend on MANES and when ad hoc WiFi (or other solutions like QualComm's AllJoyn [83]) is available on stock Android phones, Shout will run on them as well.

There are two other practical difficulties with deploying ad hoc WiFi.

First, many users employ WiFi to connect to an access point and the Internet. Using the WiFi card in ad hoc mode instead is not acceptable. Time-multiplexing methods exist to connect to two network "simultaneously" [84–87], but these are not yet widely deployed. The emerging WiFi-Direct standard [72] is new peer-to-peer technology based on WiFi that

is supported by Android. Some WiFi drivers do support simultaneous use of WiFi-Direct and an access point, so we hope simultaneous support for the ad hoc is forthcoming as well[16].

Second, average power consumption is much higher in ad hoc mode. When connected to an access point, WiFi transceivers may sleep most of the time, allowing the AP to buffer incoming packets and waking up only occasionally to check this queue. In ad hoc mode, no buffering access point exists and the device must remain listening at all times. This problem is solvable—for instance, by synchronizing the sleep schedules of the ad hoc devices—but is beyond the scope of Shout. Solutions needs to be incorporated into the WiFi protocols and device drivers. As with dual use, we hope the emergence of WiFi-Direct leads this charge.

[16] An alternative is to use WiFi-Direct, instead of ad hoc WiFi for Shout. This direction looks promising, but we have not pursued it yet.

CHAPTER 4

Mobile Ad Hoc Network Emulation System

4.1 Introduction

For ad hoc network applications like Whisper and Shout, the human participants strongly affect performance. As device carriers, their motions define the topological characteristics of the network, determining connectivity and influencing throughput and latency. As the users of the applications, their interactions and interests determine the ideal information flow—what should be sent where and when. Accurate consideration of these human properties is paramount for ad hoc system design and optimization.

Testing and characterization is commonly done through simulation—model- or trace-based—or small-scale deployment, but these approaches suffer some limitations. Models of human motion don't capture the nuance of the real world and detailed models of human–application interaction simply do not exist. Traces can provide finer granularity, but cannot capture the influence of modifications to the application on the human behavior. Deployment with real people is much better, but is usually limited to small groups. Specialized hardware and platform software impose significant per-participant costs. Larger-scale deployments are needed.

The ubiquitous smartphone appears an excellent avenue for large-scale deployments, as the hardware is already paid for, distributed, and in everyday use. Unfortunately, Android, the most popular smartphone platform, disables the ad hoc functionality of the included

WiFi chipset, crippling its use. Further, the WiFi transceiver is usually employed for Internet access and thus not available for ad hoc use[1]. Additionally, the WiFi ad hoc mode does not use the same power-saving tricks as infrastructure mode and thus has higher power consumption[2].

Using existing smartphones still appears the easiest and cheapest path for large-scale deployments, so we have built MANES, a mobile ad hoc network emulation system that works around the aforementioned issues with ad hoc connectivity. The system estimates the network topology using sensor readings provided by the client devices and relays packets through the infrastructure network—WiFi access points or cellular towers—to the estimated, in-range neighbors. Other efforts, like QualComm's AllJoyn[3], tackle a similar problem, but are designed for long-term production deployments, not research. That approach, based on peer-to-peer networking, does not offer the same controllability and observability as MANES.

We make the following primary contributions[4].

- We describe MANES, an emulation system for ad hoc 802.11 that allows researchers to run their protocols and applications on commodity smartphones, enabling large-scale deployments at low cost.

- We provide a production implementation of the MANES server and an Android client. The client could be easily ported to other platforms as well. This software will be released to the research community.

- We develop a technique to estimate ad hoc connectivity from the signal strengths of the access points visible to both devices.

[1] Solutions based on time-multiplexing allowing "simultaneous" connect to multiple APs or ad hoc networks exist [84, 85], but none ship with commodity smartphones.

[2] Solutions are possible here too, but would not be available on existing commodity devices.

[3] http://www.alljoyn.org

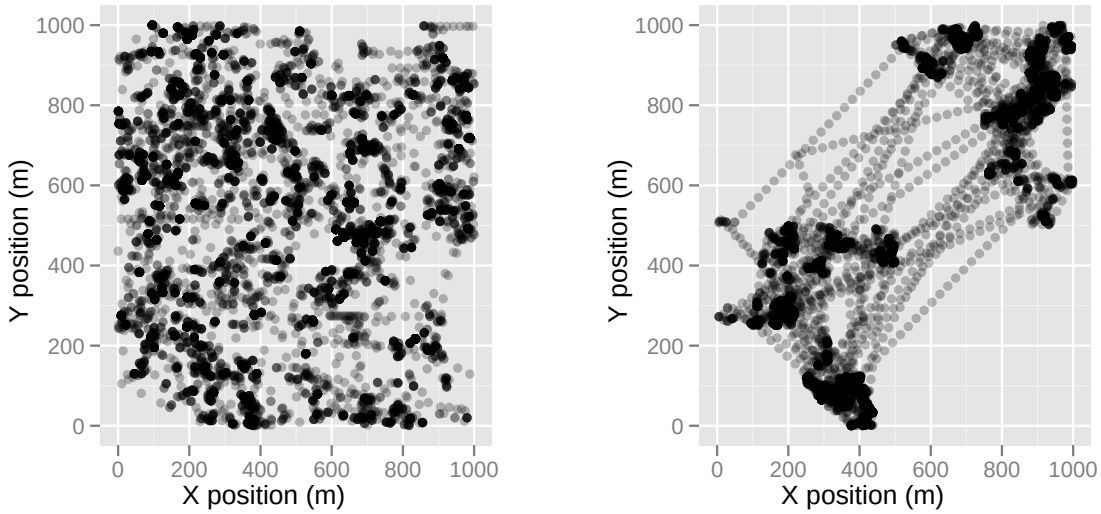
[4] MANES is very much a collaborative effort. In particular, David Adrian, Yue Liu, and Gulshan Singh contributed to the implementation. Yue Liu and Rongrong Tao designed and implemented the topology estimation. All portions of the system are described in this chapter for completeness.

4.2 Difficulties with Mobility Models or Why MANES?

When human motion patterns are the primary concern, mobility model-based simulations may appear sufficient. Selecting an appropriate model is necessary—random waypoint is clearly insufficient [88]—but much work has gone into developing such models and many have been proposed [3, 4, 50, 89, 90]. Despite this plentiful supply, selecting one appropriate for a given simulation is still rather difficult. Many of the models are incomparable (i.e., they model different features of human mobility), so a single "correct" model does not exist and selecting an appropriate one is difficult.

Current human mobility models each attempt to capture various statistical features of human motion determined from motion traces for real humans. Consequently, the models are distinguished in two primary ways: (1) by the qualitative set of features modeled (e.g., distribution of flight lengths) and (2) the quantitative fits (e.g., power law distribution with $\alpha = 2$) for those features, usually inferred from human traces. Figure 4.1 illustrates the qualitative difference by showing the spatial density of nodes for traces from two mobility models, TLW [3], which does not model the "hotspot" nature of human locations, and SLAW [4], which does.

The quantitative differences are more nuanced, as they depend on the traces to which a model was "fit" and, consequently, are influenced by any biases (intentional or accidental) in the trace populations and measurement methods. Figure 4.2 illustrates this for aggregate (i.e., population, not single individual) flight length distributions derived from three populations: (1) fine-grained traces (from GPS) for students on a university campus [4], (2) coarse-grained traces (from cell-tower locations during calls) for two populations of cell-phone users in Europe—a set of 100,000 users and a subset of 10,000 users chosen for their frequent and regular calling activity— [46] and (3) coarse-grained traces (from airline ticket data) for United States travelers. All four capture the long-tailed nature of human movement, but with three different distributions (power-law, power-law with exponential cut-off, and exponential, respectively) and, for the two cell-phone user populations, the same cut-off



(a) TLW: no "hotspot"

(b) SLAW: "hotspots"

Figure 4.1: Example node spatial distributions (over 20 individual traces) from the TLW [3] and SLAW [4] models. SLAW captures the notion of "hotspots" in human locations, while TLW does not.

power-law distribution but with different parameters.

In theory, with enough fine-grained traces, a more comprehensive model could be developed. Unfortunately, obtaining such traces is difficult; privacy concerns (and, before the proliferation of GPS-equipped smartphones, technical and economic constraints) preclude the collection and distribution of fine-grain, long-duration spatio-temporal traces for large sample sets. Instead, the traces used for modeling are biased by reducing spatial and temporal resolution, Thus, choosing an appropriate model requires determining both the desired qualitative features and their quantitative instantiations, selecting one whose underlying data-set properly captures them.

These differences raise several concerns for those using the models. How should one select a model? How does one determine if the model correctly captures the behaviors on which the one's protocols are sensitive, especially when the model is used to discover those behaviors? How much confidence should be placed in the result?

These difficulties motivated us to pursue direct, deployment-based characterization.

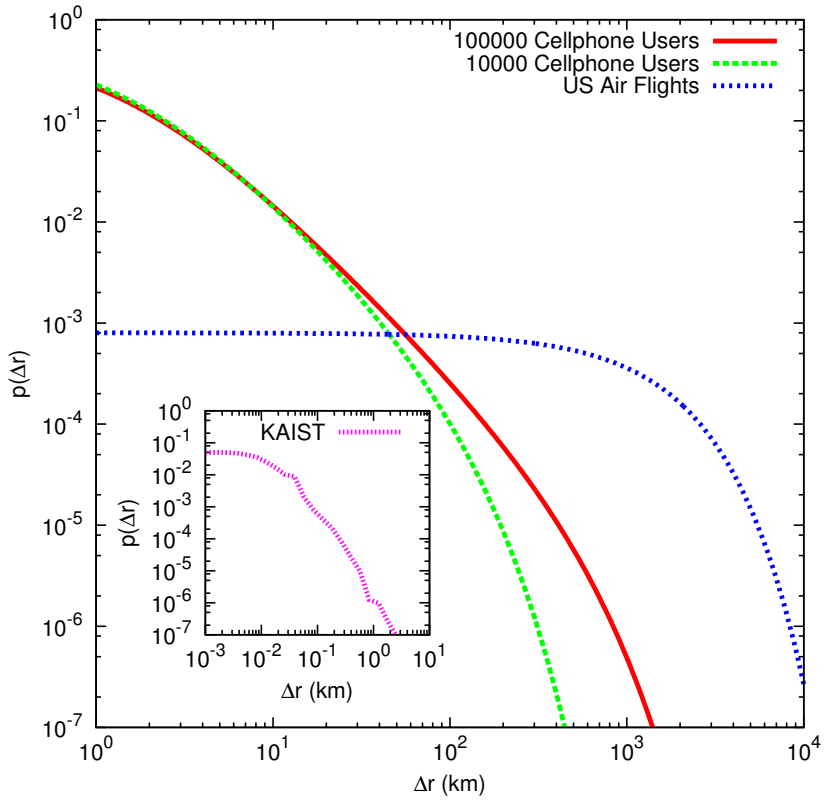


Figure 4.2: Flight length probability density functions for four different data sets, illustrating their underlying biases.

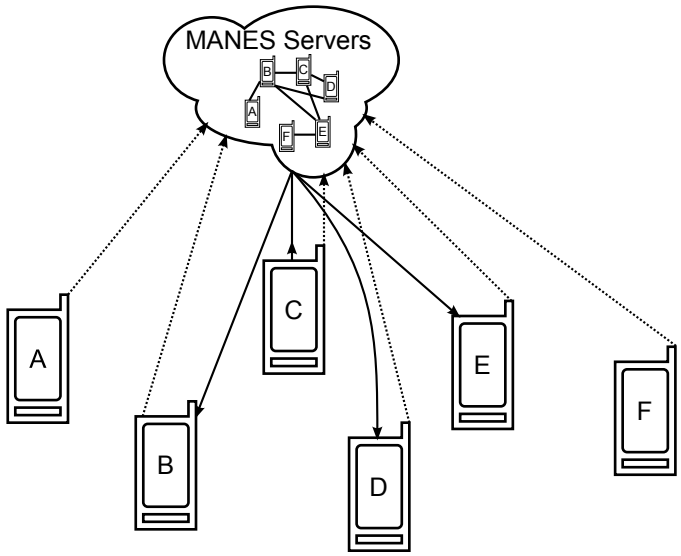


Figure 4.3: Overview of MANES architecture. All clients report GPS and WiFi observations, which are used to form an estimated topology. Packets are relayed via MANES, according to this estimate. In the example, device C broadcasts a packet that is relayed to B, D, and E.

MANES is the result. Simulation is still valuable—as discussed later, MANES does not accurately model congestion, collisions, or the detailed network timing—but for applications where human behavior is the primary independent variable, we believe deployment is not just better, but necessary.

4.3 Architecture

This section describes the design considerations and architecture of MANES.

4.3.1 Architecture Overview

Figure 4.3 illustrates the portions of the operation of MANES, topology estimation and packet relaying.

Client devices report the observed signal strengths of visible WiFi access points and, when available, the location reported by GPS. The server analyzes these reports to estimate

the ad hoc network topology. Intuitively, many devices that can see the same access points will be within ad hoc range. Similarly, devices determined to be in close proximity by GPS should be within ad hoc range. The methods used to compute scalar link qualities from these readings are described in Section 4.4.

To broadcast a packet, a client first sends it to the MANES server. The server relays the packet to each connected device in the estimated topology. Packets are delivered to clients via UDP, but some may sit behind NATs or firewalls. Periodic keepalives are sent to the server to keep the NAT mapping or firewall hole open.

4.3.2 Problem Domain

MANES is intended for deploying and testing low throughput ad hoc applications dependent on human motion and interaction. Packets are relayed through a MANES server, over cellular and WiFi connections with highly variably latency and bandwidth. The protocols (and subsequent analysis) should

- tolerate occasional delivery latencies of up to several seconds, as packets traverse cellular networks [91],
- have little total throughput, as cellular plans are often limited to several gigabytes/month or less,
- be insensitive to collision or congestion control, as these are not modeled (most protocols meeting the throughput requirement will meet this as well), and
- be robust to inaccuracies in topology estimation, for device pairs that are borderline.

Whisper and Shout—our applications of interest—both fit this category, but low-level protocols may not. For example, those attempting to maximize throughput or reduce collisions in indoor environments are not good candidates for MANES.

4.3.3 Desired Properties and Design Challenges

An emulation system for ad hoc network researchers should not just emulate topologies, but also help researchers and developers create, deploy, and test their applications, eventually migrating them to true ad hoc support. As such, we desire the following properties for MANES.

- **Accuracy:** We attempt accurate estimation of link qualities, as this implies accurate topology. As will be discussed, such estimation can be difficult. When signals are strong or non-existent, MANES does well. Weak signals are estimated more poorly.
- **Scalability:** The system should handle hundreds or thousands of users, for large-scale deployments. MANES is built to be horizontally-scalable, so the limiting factor is server and bandwidth costs. Delivery latencies scale with neighborhood density, which are usually bounded.
- **Usability:** Developers should find the system easy-to-use. MANES provides a simple API—standard send() and receive()—methods, so applications can be easily transitioned to other underlying transmission protocols, like pure ad hoc WiFi.
- **Efficiency:** Smartphone battery capacity is limited and generally needs to last at least a day. On the author's phone, MANES uses just 4% of battery capacity over a typical 10 hour work day[5].
- **Observability:** Researchers need access to topology and transmission histories to characterize performance and possibly drive trace-based simulations. MANES logs topology changes, the raw data used to estimate the topology, and all transmissions in flat textual files for easy analysis.
- **Controllability:** Researchers may want to modify the network topology, inject transmissions, or block others to study various effects. The centralized MANES architec-

[5]Energy usage increases with the traffic rate. Most of that 4% is attributable to the topology estimation, not packet transmission.

ture makes this easy. As an example, with Shout, we employ "virtual" reshouters to increase the effective density and improve connectivity during early adoption.

Satisfying these properties requires solving two primary technical challenges.

Accurate Topology Estimation: Inferring the WiFi link quality between two devices from indirect measurements is non-trivial. We take a two-pronged approach. First, when both devices can observe some WiFi access points, we use the signal strengths of the APs visible to both devices. Intuitively, if both see distinct sets of APs, they are not within WiFi range of each other. If both see the same set, they are likely nearby and within range. The actual AP signal strengths are used to estimate a more-detailed scalar link quality, as described in Section 4.4. On the assumption that if no WiFi access points are visible, then devices are outdoors, the link quality is determined by the distance between them, as reported by GPS.

Energy Efficient Topology Updates: The topology must be up-to-date to ensure packets are delivered to the correct devices. Reporting the data used for estimation is consumes energy on the clients, so the topology update latency and client energy usage must be balanced. The more-efficient WiFi scans are used to detect motion. This scan runs periodically, and only when the results change is the power-hungry GPS turned on. Between GPS readings, a simple velocity model is used to predict the current location.

4.3.4 Design Choices

API: The interface provided to application and protocol developers influences how easily applications can be switched between MANES and other transport mechanisms[6]. MANES is intended to emulate the 802.11 ad hoc broadcast mode, a layer 2 networking protocol, so we mimic that interface [92]. The frame includes both the packet contents and an L3 protocol identifier, indicating which protocol or application should handle the incoming

[6]Systems like AllJoyn provide much higher abstractions to application developers, reducing working at the risk of platform lock-in.

packet. Unlike typical network stacks, the receive method is a blocking and called by the application, not a callback initiated by MANES. This approach is more familiar to application developers used to the socket abstraction.

`register(int protocolId)` registers the application for a particular protocol id.

`send(byte[] packet)` broadcasts a frame with packet contents. The protocol id specified during registration is used.

`byte[] receive()` awaits (blocks for) the next incoming packet for the registered protocol id.

Scalability: Topology estimation should be fast, regardless of the number of users. Consider that naïve topology estimation would require $O(n^2)$ comparisons. MANES scales through two techniques, a horizontal, distributed architecture to spread load and an efficient $O(1)$ topology estimation algorithm[7].

The distributed architecture is backed by a horizontally-scalable, key-value database, *Voldemort* [93], an implementation of Amazon's Dynamo architecture [94]. All MANES servers are stateless, so requests from multiple clients are easily spread among them. Topology estimation is performed by the server handling the upload of new WiFi or GPS readings. Thus, the latency for the topology update is independent of the client upload rate; it depends solely on the time for one topology computation.

Topology estimation is kept efficient by only comparing AP signal strengths or locations with those nearby. The database is used as a large hash-based index for this purpose. A reverse mapping from AP to client id is maintained, allowing quick look up of all clients that reported observations for a given AP. Similarly for GPS, the Earth's surface is divided into 250 m by 250 m "squares" and a reverse mapping from grid to clients stored, allowing fast retrieval of all clients within 250 m of a given location[8]. With this approach, the computation time is independent of network size, scaling instead with network density.

[7] The topology estimation algorithm is actually $O(d^2)$, but d, the network density, is bounded.

[8] All clients in the nine squares surrounding the location must be checked for proximity. This process is still constant time and reasonably quick, although it could be reduced through a more precise indexing scheme.

Packet Delivery: Packet delivery from server to smartphone client is, unfortunately, non-trivial. The NAT and firewalls guarding many networks prevent devices from accepting incoming messages from unknown sources—all connections must be initiated by the device and only responses are allowed through. Instead, a persistent connection—like an open TCP connection—to the server and initiated by the client is needed.

Managing large numbers of TCP connections takes care, because they are stateful, occupying resources like memory and port numbers. Further, on smartphones that frequently drop connections as network connectivity changes, they must be carefully monitored and restarted. MANES does not require the ordering and reliability guarantees of TCP, so to simplify, we use the stateless UDP protocol.

Each client periodically (e.g., every 30 seconds) send a UDP packet to the MANES server. The IP address and port number are stored in the database and used when relaying any packets. This system consumes no resources on the servers (just in the scalable database) is implicitly resilient to changes in network state. responds to changes in network connectivity.

4.3.5 Client Architecture

Figure 4.4 shows the architecture of the client software, implemented for Android. Most components run in background service.

Location Tracker: The location tracker collects the information needed for topology estimation, i.e., WiFi access points signal strengths and GPS readings. When the readings have changed, it uploads them to the MANES server so the topology can be recomputed.

Packet Manager: The packet manager accepts incoming frames from the MANES server and routes the contained packet to the appropriate application by protocol id. Frames with no registered application are dropped. The packet manager also takes packets from the applications, forwarding them to the MANES server for broadcast.

Keepalive Manager: The keepalive manager sends the UDP keepalive packet every 30 seconds. Keepalives are also sent whenever the network connectivity changes—e.g., from

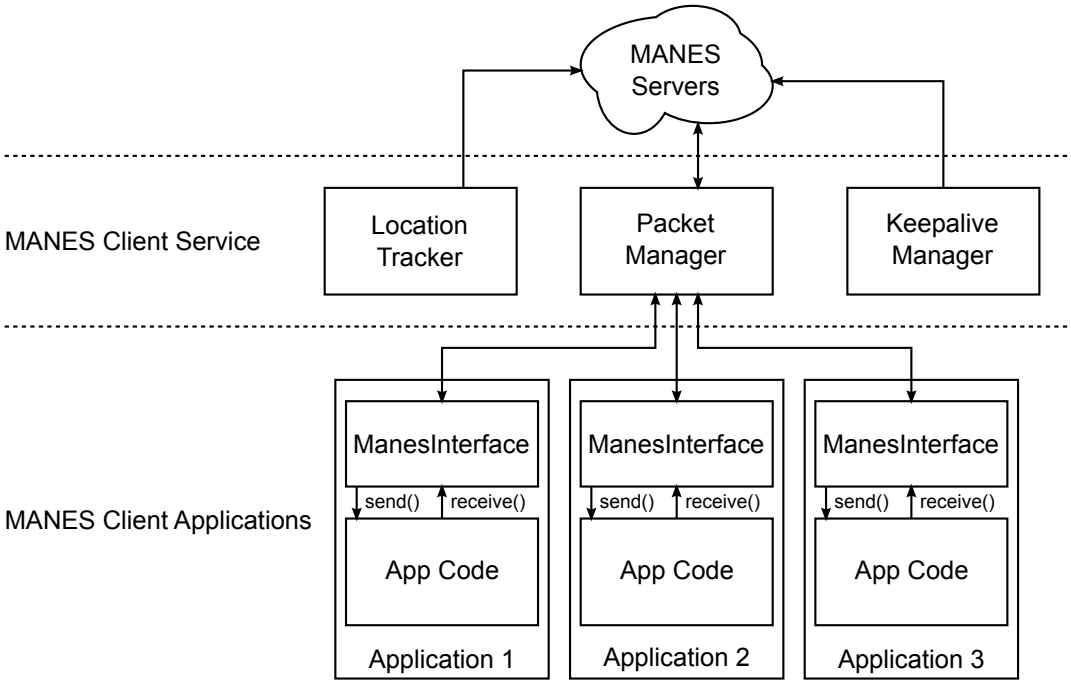


Figure 4.4: Architecture of MANES client software.

WiFi to cellular—to minimize down time.

ManesInterface: Each application instantiates its own instance of ManesInterface, which provides the **register()**, **send()**, and **receive()** methods mentioned earlier. This gives developers simple API, masking the complexity of communicating with a background service in Android.

4.3.6 Server Architecture

Figure 4.5 shows the architecture of the server.

Topology Database: The topology database is multi-server Voldemort cluster that stores the current topology, past WiFi and GPS readings needed for future topology estimations, and the hash-based indexes needed for efficient topology estimation.

Topology Estimator: The topology estimator servers accept location and WiFi scan reports from clients, store them in the database, and then compute a new topology estimate. Details of the topology estimation methods are given in the next section.

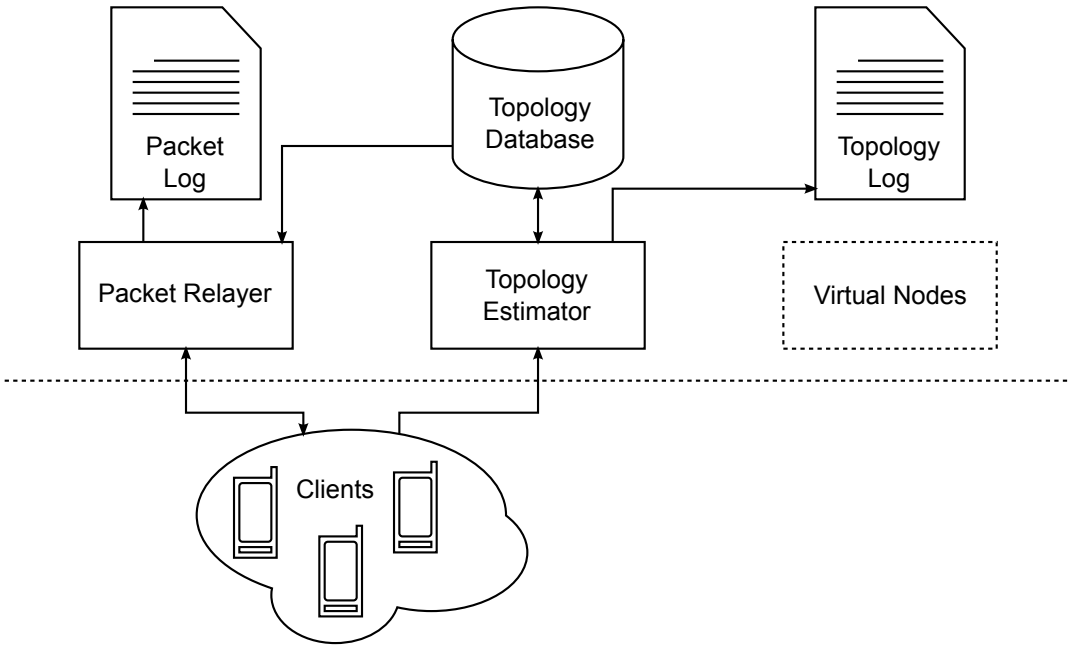


Figure 4.5: Architecture of MANES server system.

Packet Relayer: The packet relay servers accept incoming packets, relaying them to the clients in-range according to the current topology estimate. It also handles the keepalive packet, storing the current IP address and port number of the client in the database.

Text-based Logs: All information, namely raw GPS and WiFi scan reports, estimated topologies, and sent packets are logged into text-based logs for observability and later analysis.

Virtual Nodes: The virtual nodes represent the option for controllability. This could be a component that mimics real clients, increasing effective density, or a component that injects specific content to study the impact. MANES is intentionally modular, so researchers introduce the specific control they need.

4.4 Topology Estimation

MANES estimates the link quality between devices to build the topology—if these link estimates are accurate, so is the overall topology. However, determining the link quality

from the limited sensors on a smartphone is non-trivial. In this section, we summarize our approach[9], which uses WiFi access point scan results and GPS readings.

We use *Packet Reception Rate* (PRR), the fraction of sent packets successfully received, to quantify link qualities. MANES randomly drops packets with probability proportional to $1 - \mathrm{PRR}$ to model this effect. PRR cannot be directly determined from a single measurement (it's an average over multiple transmissions), so we instead look for measurements from which to estimate it.

The Received Signal Strength (RSS or RSSI), a value reported by commodity WiFi devices, fits the bill[10]. Experiments with Nexus One smartphones, rooted and modified to support ad hoc WiFi, were used to determination the ground truth PRR. Pairwise RSSI and PRR measurements were made in a 110 locations spread throughout a large academic building, including multiple floors, hallways, offices, labs, and open spaces. RSSI correlates well with PRR at the extremes, i.e., when the signal is weak and PRR is zero or when the signal is strong and the PRR is one. The prediction accuracy is worse in the transition zone—−90–−80 dBm—but is still acceptable. We use the prediction function $PRR \approx 1 - \exp(\frac{-RSSI - 97.22}{4.16})$.

4.4.1 Received Signal Strengths of Visible WiFi Access Points

WiFi Direct is a new technology and not yet widely supported. Thus, most devices cannot use it to directly measure RSSI. Instead, we develop a method to estimate RSSI from those of visible access points. Intuitively, devices observing the same APs should be physically close and thus able to communicate directly. Further, observed AP signal strengths should be correlated with the attenuation of that wireless environment and thus reveal something about the inter-device signal strength.

[9]This method was primarily developed and analyzed by Yue Liu with the help of Rongrong Tao, but is summarized here for completeness.

[10]The *Signal-to-Interface plus Noise Ratio (SINR)* is better proxy for PRR [95], but is not calculated or reported by commodity hardware.

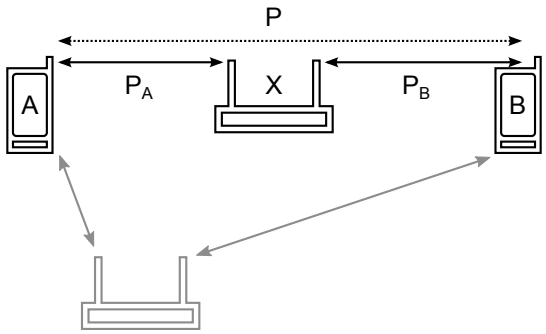


Figure 4.6: Heuristic for estimating the signalstrength P between two devices from observed APs.

Figure 4.6 illustrates the heuristic approach, based on this intuition. Imagine that an access point X was on the straight line between two devices, A and B. The inter-device attenuation can be estimated as the sum of the two device-to-AP attenuations. Thus, the inter-device RSSI, P can be estimated from the two RSSI observations of the AP, P_A and P_B:

$$P = -10r \cdot \log\left(10^{\frac{P_A}{10r}} + 10^{\frac{P_B}{10r}}\right), \tag{4.1}$$

where r is an assumed path loss exponent shared by all paths.

In reality, an AP is unlikely to sit directly on that line. Instead, we heuristically choose the AP closest to that line, i.e., the one indicating the strongest power:

$$\bar{P} = \max_i \left[-10r \cdot log\left(10^{\frac{P_{i,A}}{10r}} + 10^{\frac{P_{i,B}}{10r}}\right)\right]. \tag{4.2}$$

This technique accurately predicts PRR when the signals are strong ($PRR = 1$) or non-existent ($PRR = 0$), but lacks the fidelity to determine intermediate PRRs when connectivity is borderline.

4.4.2 GPS Distance Measurement

Some environments, particular those outdoors, may not have enough access points to support RSSI estimation. In these scenarios, we instead fall back to location-based PRR estimation

using GPS. Distance and PRR is not as highly correlated as RSSI and PRR, but is still sufficient, particularly for outdoor devices. WiFi-based estimates are preferred; GPS is used only when WiFI is unavailable.

CHAPTER 5

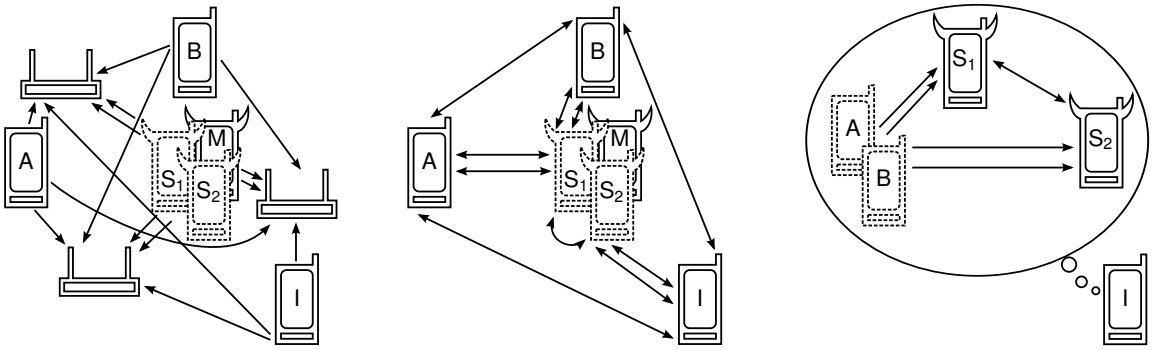
Mason Test

5.1 Introduction

The open nature of wireless ad hoc networks (including delay-tolerant networks [96]) enables applications ranging from collaborative environmental sensing [97] to emergency communication [98], but introduces numerous security concerns since participants are not vetted. Solutions generally rely on a majority of the participants following a particular protocol, an assumption that often holds because physical nodes are expensive. However, this assumption is easily broken by a Sybil attack. A single physical entity can pretend to be multiple participants, gaining unfair influence at low cost [57]. Newsome et al. survey Sybil attacks against various protocols [99], illustrating the need for a practical defense.

Proposed defenses (see Levine et al. for a survey [100]) fall into two categories. *Trusted certification* methods [101, 102] use a central authority to vet potential participants and thus are not useful in open ad hoc (and delay-tolerant) networks. *Resource testing* methods [103–106] verify the resources (e.g., computing capability, storage capacity, real-world social relationships, etc.) of each physical entity. Most are easily defeated in ad hoc networks of resource-limited mobile devices by attackers with access to greater resources, e.g., workstations or data centers.

One useful class of defenses is based on the natural spatial variation in the wireless propagation channel, an implicit resource. Channel responses are uncorrelated over distances



(a) RSSI observations from trusted APs identity the Sybils, S_1 and S_2, from attacker M.

(b) In ad hoc networks, the participants themselves act as observers, but can maliciously report falsified values.

(c) If I believes the falsified observations from S_1 and S_2, it will incorrectly accept them and reject A and B as Sybil.

Figure 5.1: Prior work [5, 6] assumes trusted RSSI observations, not generally available in ad hoc and delay-tolerant networks. We present a technique for a participant to separate true and false observations, enabling use in ad hoc networks. (Arrows point from transmitter to observer.)

greater than half the transmission wavelength [107] (6.25 cm for 2.4 GHz 802.11), so two transmissions with the same channel response are likely from the same location and device [5, 108]. However, the existing Sybil defenses in this class are not directly usable in open ad hoc networks of commodity devices.

Xiao et al. observe that in OFDM-based 802.11 the coherence bandwidth is much smaller than the system bandwidth and thus the channel response estimates at well-spaced frequency taps are uncorrelated, forming a vector unique to the transmitter location and robust to changes in transmitter power [5]. Unfortunately commodity 802.11 devices do not expose these estimates to the driver and operating system, restricting this technique to specialized hardware and access points.

Commodity devices do expose an aggregate, scalar value, the received signal strength. RSSI is not robust to changes in transmitter power, so a vector of observations from multiple receivers—a *signalprint*—is used instead. Several authors have proposed such methods [5, 6, 109–113] assuming trusted, true observations. In open ad hoc networks, observations are untrusted, coming from potentially-lying neighbors, as illustrated in Figure 5.1. Trust-less

methods have been proposed, but have various limitations (e.g., devices must be non-mobile [114], colluding attackers can defeat the scheme [115], or are limited to outdoor environments with predictable propagation ranges [116]). Instead, a general method to separate true and false observations is needed.

We make two observations that enable separation. First, with high probability attackers cannot produce false observations that make conforming identities look Sybil. Second, nodes complying with the protocol outnumber physical attacking nodes (motivating the Sybil attack), implying that most non-Sybil identities tell the truth.

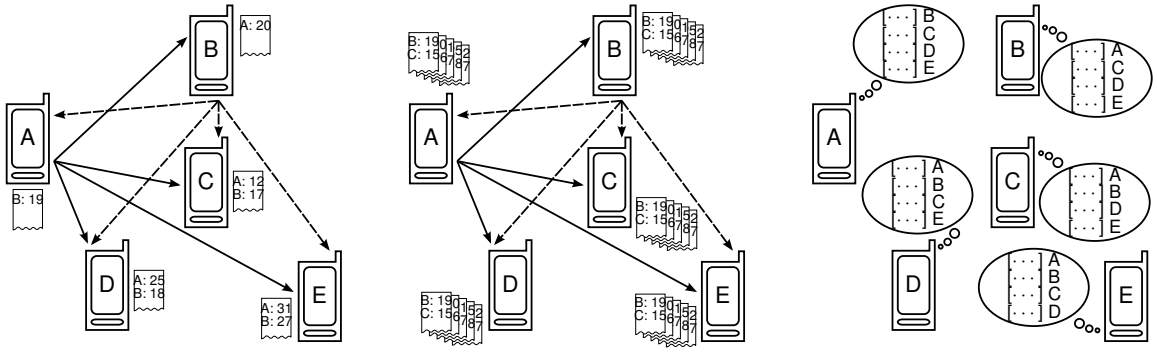
Most past work assumes nodes are stationary, as moving attacks can easily defeat signalprint-based detection. As noted, but not pursued, by Xiao et al., successive transmissions from the same node should have the same signalprint and attackers likely cannot quickly (i.e., in milliseconds) switch between precise positions [5]. We develop a challenge–response protocol from this idea and study its performance on real deployments.

We make the following primary contributions[1].

- We prove conditions under which a participant can separate true and false RSSIs reported by untrusted neighbors, enabling signalprint-based Sybil detection in ad hoc networks of mutually distrusting nodes.
- We develop an $O(n^3)$ algorithm for this separation suitable for networks with hundreds of one-hop neighbors.
- We develop a challenge-response protocol to detect attackers using motion to bypass the signalprint-based Sybil defense.
- We describe the Mason test, a practical protocol for Sybil defense based on these ideas. We implemented the Mason test as a Linux kernel module for 802.11 ad hoc networks[2] and characterize its performance in real-world scenarios.

[1] This work was performed in close collaboration with Yue Liu. She identified the initial problem, developed an initial solution which served as a foundation for the one described within, and performed some of the evaluation.

[2] `http://github.com/EmbeddedAtUM/mason/`



(a) Nodes record their observed RSSIs of probes broadcast by neighbors. A and B have sent; C, D, and E are next.

(b) RSSI observations are shared among all participants. Malicious nodes could lie about their observations.

(c) Each participant selects a subset of the observations to form signalprints for Sybil detection.

Figure 5.2: The solution framework for signalprint-based Sybil detection in ad hoc networks. This chapter fleshes out this concept into a safe and secure protocol, the Mason test.

5.2 Problem Formulation and Background

In this section, we define our problem, overview the solution framework, describe our attack model, and briefly review the signalprint method.

5.2.1 Problem Formulation

Our high-level goal is to allow a wireless network participant to occasionally determine which of its one-hop neighbors are non-Sybil. These identities may safely participate in other protocols. In mobile networks, the process must be repeated occasionally (e.g., once per hour) as the topology changes. Safety is more important than system performance, so nearly all Sybil identities should be identified, but some non-Sybils may be rejected.

Prior work showed the effectiveness of signalprint techniques with trusted RSSI observations. We extend those methods to work without a priori trust in any observation. As illustrated in Figure 5.2, we assume an arbitrary identity (or condition) starts the process. Participants take turns broadcasting a probe packet and recording the observed RSSIs. These observations are then shared, although malicious nodes may lie. Each participant

individually selects a (hopefully truthful) subset of identities for signalprint-based Sybil classification.

This paper presents our method for truthful subset selection and fleshes out this framework into a usable, safe, and secure protocol. As with any system intended for real-world use, we had to carefully balance system complexity and potential security weaknesses. Section 5.9 discusses these choices and related potential concerns.

5.2.2 Attack Model

We assume attackers have the following capabilities and restrictions.

1. Attackers may collude through arbitrary side channels.

2. Attackers may accumulate information, e.g., RSSIs, across multiple rounds of the Mason test.

3. Attackers have limited ability to predict RSSI observations of other nodes, e.g., 7 dBm uncertainty (see Section 5.5), precluding fine-grained pre-characterization.

4. Attackers can control transmit power for each packet, but not precisely or quickly steer the output in a desired direction, e.g., beam-forming.

5. Attackers cannot quickly and precisely switch between multiple positions, e.g., they do not have high-speed, automated electromechanical control.

These capabilities and restrictions model attacking nodes that are commodity devices, a cheaper attack vector than distributing specialized hardware. These devices could be obtained by compromising those owned by normal network participants or directly deployed by the attacker.

One common denial-of-service (DOS) attack in wireless networks—jamming the channel—cannot be defended against by commodity devices. Thus, we do not defend against other more-complicated DOS attacks.

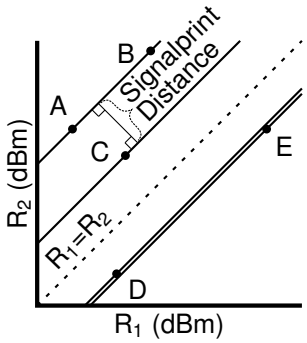


Figure 5.3: Sybils, A–B and D–E, occupy nearby slope-1 lines.

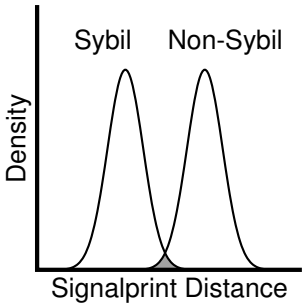


Figure 5.4: The distance threshold trades false positives for negatives.

Notably, we assume attackers do not have per-antenna control of MIMO (Multiple-Input and Multiple-Output) [117] devices. Such control would defeat the signalprint method (even with trusted observers), but is not a feasible attack. Commodity MIMO devices (e.g., 802.11n adapters) do not expose this control to software and thus are not suitable attack vectors. Distributing specialized MIMO-capable hardware over large portions of the network would be prohibitively expensive.

We believe that the signalprint method can be extended to MIMO systems (see our technical report for an overview [118]), but doing so is beyond the scope of this work. Our focus is extending signalprint-based methods for ad hoc networks of commodity devices by removing the requirement for trusted observations.

5.2.3 Review of Signalprints

We briefly review the signalprint method. See prior work for details [5, 109]. A *signalprint* is a vector of RSSIs at multiple observers for a single transmission. Ignoring noise, the vector of received powers (in logarithmic units, e.g., dBm) at multiple receivers for a given transmission can be modeled [107] as $\vec{\mathbf{s}} = \vec{\mathbf{h}} + p\vec{\mathbf{1}}$, where p is the transmit power and $\vec{\mathbf{h}}$ is the attenuation vector, a function of the channel amplitude response and the receiver characteristics. Transmissions from different locations have uncorrelated signalprints, as the channel responses are likely uncorrelated. Those from the same location, however, share

a channel response and will be correlated. That is, for two transmissions a and b from the same location with transmit powers p_a and $p_b = p_a + c$, the signalprints $\vec{\mathbf{s}}_\mathbf{b} = \vec{\mathbf{h}} + p_a\vec{\mathbf{1}}$ and $\vec{\mathbf{s}}_\mathbf{b} = \vec{\mathbf{h}} + (p_a + c)\vec{\mathbf{1}}$ are related as $\vec{\mathbf{s}}_\mathbf{b} = \vec{\mathbf{s}}_\mathbf{a} + c\vec{\mathbf{1}}$.

This is illustrated geometrically in Figure 5.3 for a two-receiver signalprint. A and B are Sybil, while C is not. D and E are also Sybil, but due to noise the signalprints are not perfectly correlated. Instead, signalprints occupying lines closer than some threshold are taken to be Sybil.

Definition. The *signalprint distance* $\mathrm{d}(\vec{\mathbf{s}}_\mathbf{a}, \vec{\mathbf{s}}_\mathbf{b})$ between two signalprints $\vec{\mathbf{s}}_\mathbf{a}$ and $\vec{\mathbf{s}}_\mathbf{b}$ is the perpendicular distance between the slope-1 lines containing them. Letting

$$\vec{\mathbf{w}} \triangleq \vec{\mathbf{s}}_\mathbf{a} - \vec{\mathbf{s}}_\mathbf{b}$$

be the distance vector between the signalprints and

$$\vec{\mathbf{v}}_\perp \triangleq \vec{\mathbf{w}} - \frac{\vec{\mathbf{w}} \cdot \vec{\mathbf{1}}}{\|\vec{\mathbf{1}}\|^2}\vec{\mathbf{1}}$$

be the vector rejection of $\vec{\mathbf{w}}$ from $\vec{\mathbf{1}}$, then

$$\mathrm{d}(\vec{\mathbf{s}}_\mathbf{a}, \vec{\mathbf{s}}_\mathbf{b}) = \|\vec{\mathbf{v}}_\perp\|.$$

As shown in Figure 5.4, the distance distributions for Sybil and non-Sybil identities overlap, so the threshold choice trades false positives for negatives. A good threshold can detect at least 99.9% of Sybils while accepting at least 95% of non-Sybils [5, 109].

5.3 Sybil Classification From Untrusted Signalprints

We describe our method to detect Sybils using untrusted RSSI observations. No general solution exists, so we derive sufficient, likely conditions that enable classification.

5.3.1 Power of Falsified Observations

Signalprints contain observations from multiple observers (4–6 for reasonable accuracy [6]). Since a node trusts only its own observations, those from other observers are untrusted. Consider how falsified RSSI observations can influence Sybil detection. First, one can easily construct false observations to make a Sybil identity look non-Sybil. To see this, recall that two identities are considered Sybil only if all observers report the same RSSI difference. Randomly chosen values will almost certainly not satisfy this condition. The second, making non-Sybils look Sybil, is much harder. The RSSI difference is fixed by the initiator's trusted self-observation, so an attacker would have to learn or guess this difference. The method described in this section relies on this difficulty, which is quantified in Section 5.5.

5.3.2 Terminology

I is the set of participating identities. Each is either Sybil or non-Sybil and reports either true or false[3] RSSI observations, partitioning the identities by their Sybilness (sets S and NS) and the veracity of their reported observations (sets T and L).

	S	NS
L	LS	LNS
T	TS	C

Truthtelling, non-Sybil identities are called *conforming* (set C). Liars and Sybil identities are called *attacking* (sets LS, LNS, and TS). Our goal is to distinguish the S and NS partitions using the reported RSSI observations without knowing a priori the L and T partitions.

Definition. An *initiator* is the node performing Sybil classification[4]. It trusts its own RSSI observations, but no others.

[3] A reported RSSI observation is considered *false* if signalprints containing it misclassify some identities.
[4] All participants perform classification individually, so each is the initiator in its own classification session.

Definition. A *receiver set*, denoted by R, is a subset of identities ($R \subseteq I$) whose reported RSSI observations, with the initiator's, form signalprints. Those with liars ($R \cap L \neq \emptyset$) produce incorrect classifications and those with only truthtellers ($R \subseteq T$) produce the correct classification.

Definition. A *view*, denoted by V, is a classification of identities as Sybil and non-Sybil. Those classified as Sybil (non-Sybil) are said to be Sybil (non-Sybil) *under* V and are denoted by the subset V_{S} (V_{NS}). A view V obtained from the signalprints of a receiver set R is *generated* by R, denoted by $R \mapsto V$ (read: R generates V), and can be written $V(R)$. Identities in R are considered non-Sybil, i.e., $R \subseteq V_{\mathrm{NS}}(R)$. A *true view*, denoted by $\overline{V}$, correctly labels all identities, i.e., $\overline{V}_{\mathrm{S}} = S$ and $\overline{V}_{\mathrm{NS}} = NS$. Similarly, a *false view*, denoted by $\widehat{V}$, incorrectly labels some identities, i.e., $\widehat{V}_{\mathrm{S}} \neq S$ and $\widehat{V}_{\mathrm{NS}} \neq NS$.

Definition. Incorrectly labeling non-Sybil identities as Sybil is called *collapsing*.

Assumption. To clearly illustrate the impact of intentionally-falsified observations, we first assume that true RSSI observations are noise-free and thus always generate the true view. In Subsection 5.3.6, we extend the method to handle real-world observations containing, for example, random noise and discretization error.

5.3.3 Approach Overview

It is easy to see that a fully-general solution to our problem does not exist by noting that different scenarios can result in the same reported RSSI observations (under the symmetry of identities) and are thus indistinguishable. To illustrate, consider identities $I = \{A|B\}$ reporting observations such that

$$R \subseteq A \mapsto V^1 = \{V^1_{\mathrm{NS}} = A | V^1_{\mathrm{S}} = B\} \text{ and}$$
$$R \subseteq B \mapsto V^2 = \{V^2_{\mathrm{NS}} = B | V^2_{\mathrm{S}} = A\}$$

Table 5.1: Definitions of Terms and Symbols

	Definition	**Notes**
Sets of Identities		
I	all participating identities	
NS	all non-Sybil identities	$I = \{NS \vert S\}$
S	all Sybil identities	
T	all truthful identities	$I = \{T \vert L\}$
L	all lying identities	
C	all conforming, or truthful, non-Sybil, identities	$NS = \{C \vert LNS\}$
LNS	all lying, non-Sybil identities	$S = \{TS \vert LS\}$
TS	all truthful, Sybil identities	$T = \{C \vert TS\}$
LS	all lying, Sybil identities	$L = \{LNS \vert LS\}$
V_{NS}	all identities labeled non-Sybil by view V	$I = \{V_{\mathrm{NS}} \vert V_{\mathrm{S}}\}$
V_{S}	all identities labeled Sybil by view V	
R (*receiver set*)	identities used to form signalprints	
Views		
V (*view*)	a Sybil–non-Sybil labeling of I	
$\overline{V}$ (*true view*)	a view that correctly labels all identities	$\overline{V}_{\mathrm{NS}} = NS$ and $\overline{V}_{\mathrm{S}} = S$
$\widehat{V}$ (*false view*)	a view that incorrectly labels some identities	$\widehat{V}_{\mathrm{NS}} \neq NS$ and $\widehat{V}_{\mathrm{S}} \neq S$
$V(R)$	the view generated by receiver set R	
Terms		
generates $(R \mapsto V)$	a receiver set generates a view	
initiator	node performing the Sybil classification	
collapse	classify a non-Sybil identity as Sybil	

and two different scenarios x and y such that

$$\text{in } x, \{T^x = A | L^x = B\} = I \text{ and}$$
$$\text{in } y, \{T^y = B | L^y = A\} = I.$$

Remembering that $R \subseteq T \mapsto \overline{V}$, the true view for scenario x is V^1 and for scenario y is V^2. Consequently, no method can always choose the correct view.

Since a general solution is not possible, we instead look for restricting conditions that hold in situations of practical importance and permit a method to identify the true view. In particular, we use the following two notions, formalized when needed.

- Fabricating RSSI observations that make non-Sybil identities look Sybil is difficult, so all views will correctly classify some conforming identities.
- Conforming identities outnumber lying, non-Sybils (often the very motivation for the Sybil attack).

Our approach stems from the idea that true observations, which all describe the same world, are consistent. Lies, however, are often contradict themselves. We use a notion of consistency that is quite difficult for attackers to achieve to separate the true observations.

5.3.4 View Consistency: Selecting $\overline{V}$ if $LNS = \emptyset$

This section introduces the concept of a consistent view, using the following unrealistic restriction. In Subsection 5.3.5 we lift this restriction.

Restriction 1. All liars are Sybil, i.e., $LNS = \emptyset$, and thus all non-Sybil identities are truthful, i.e., $NS \subseteq T$.

Restriction 1 endows the true view with a useful property: all receiver sets comprising the non-Sybil identities under the true view will generate the true view. We formalize this consistency as follows.

Definition. A view is *view-consistent* if and only if all receiver sets comprising a subset of the non-Sybil identities under that view generate the same view, i.e., V is view-consistent iff $\forall R \in 2^{V_{\text{NS}}} : R \mapsto V$.

Lemma 1. *Under Restriction 1, the true view is view-consistent, i.e.,* $\forall R \in 2^{\overline{V}_{\text{NS}}} : R \mapsto \overline{V}$.

Proof. Consider the true view $\overline{V}$. By definition, $\overline{V}_{\text{NS}} = NS$. By Restriction 1, $NS \subseteq T$ and thus, $\overline{V}_{\text{NS}} \subseteq T$. $\forall R \in 2^T \mapsto \overline{V}$, so $\forall R \in 2^{\overline{V}_{\text{NS}}} : R \mapsto \overline{V}$. □

Were all false views not consistent, then consistency could be used to identify the true view. A fully omniscient attacker could theoretically generate a false, consistent view by collapsing all conforming identities. However, the practical difficulty of collapsing identities prevents this. We formalize this as follows.

Condition 1. All receiver sets correctly classify at least one conforming identity, i.e., $\forall R \in 2^I : V_{\text{NS}}(R) \cap C \neq \emptyset$.

Justification. Collapsing conforming identities requires knowing the hard-to-predict initiator's RSSI observations. Section 5.5 quantifies the probability that our required conditions hold. □

Lemma 2. *Under Condition 1, a view generated by a receiver set containing a liar is not view-consistent, i.e.,* $R \cap L \neq \emptyset$ *implies* $V(R)$ *is not view-consistent.*

Proof. Consider such a receiver set R with $R \cap L \neq \emptyset$. By Condition 1, $r \triangleq V_{\text{NS}}(R) \cap C$ is not empty and since $r \subseteq C \subseteq T$, $r \mapsto \overline{V}$. By the definition of a liar, $V(R) \neq \overline{V}$ and thus R is not consistent. □

Theorem 1. *Under Restriction 1 and Condition 1 and assuming* $C \neq \emptyset$*, exactly one consistent view is generated across all receiver sets and that view is the true view.*

Proof. By Lemma 1 and Lemma 2, only the true view is consistent, so we need only show that at least one receiver set generates the true view. $C \neq \emptyset$ and thus $R = C \mapsto \overline{V}$. □

This result suggests a method to identify the true view—select the only consistent view. As Restriction 1 does not hold in practice, so we develop methods to relax it.

5.3.5 Achieving Consistency by Eliminating LNS

Consider a scenario with some non-Sybil liars. The true view would be consistent were the non-Sybil liars excluded. Similarly, a false view could be consistent were the correctly classified conforming identities excluded. If the latter outnumber the former, this yields a useful property: the consistent view for the largest subset of identities, i.e., that with the fewest excluded, is the true view, as we now formalize and prove.

Condition 2. The number of conforming identities is strictly greater than the number of non-Sybil liars, i.e., $|C| > |LNS|$.

Justification. This is assumed by networks whose protocol require a majority of the nodes to behave. In others, it may hold for economic reasons—deploying as many nodes as the conforming participants is expensive. □

Condition 3. Each receiver set either correctly classifies at least $|LNS| + 1$ conforming identities as non-Sybil or the resulting view, when all correctly classified conforming identities are excluded, is not consistent, i.e., $\forall R \in 2^I : (|V_{\text{NS}}(R) \cap C| \geq |LNS|+1) \vee (\exists Q \in 2^{V_{\text{NS}}(R) \setminus C} : V(Q) \neq V(R))$. Note that this implies Condition 2.

Justification. This is an extension of Condition 1. Section 5.5 quantifies the probability that it holds. □

Lemma 3. *Under Condition 2 and Condition 3, the largest subset of I permitting a consistent view is $I \setminus LNS$.*

Proof. $I \setminus LNS$ permits a consistent view, per Lemma 1. Let $E_R \triangleq \widehat{V}_{\text{NS}}(R) \cap C$ be the set of correctly classified conforming nodes for a lying receiver set R, i.e., $R \cap L \neq \emptyset$. $I \setminus E_R$ is the largest subset possibly permitting a consistent view under R. By Condition 3, $\forall R : |E_R| \geq |LNS| + 1$. □

Theorem 2. *Under Condition 2 and Condition 3, the largest subset of I permitting a consistent view permits just one consistent view, the true view.*

Proof. This follows directly from Lemma 3 and Theorem 1. □

In the next section, we extend the approach to handle the noise inherent in real-world signalprints.

5.3.6 Extending Consistency to Handle Noise

Noise prevents true signalprints from always generating the true view. Observing from prior work that the misclassifications are bounded (e.g., more than 99% of Sybils detected with fewer than 5% of conforming identities collapsed [5, 109]), we extend the notion of consistency as follows.

Definition. Let γ_n be the maximum fraction[5] of non-Sybil identities misclassified by a size-n receiver set. Prior work suggests $\gamma_4 = 0.05$ is appropriate (for $|C| > 20$) [5, 109].

Definition. A view is γ_n-*consistent* if and only if all size-n receiver sets that are subsets of the non-Sybil identities under that view generate a γ_n-*similar* view. Two views V^1 and V^2 are γ_n-similar if and only if

$$\left(\frac{|V^1_{\text{NS}} \cap V^2_{\text{NS}}|}{|V^1_{\text{NS}} \setminus V^2_{\text{NS}}|} > \frac{1-2\gamma_n}{\gamma_n}\right) \bigwedge \left(\frac{|V^1_{\text{NS}} \cap V^2_{\text{NS}}|}{|V^2_{\text{NS}} \setminus V^1_{\text{NS}}|} > \frac{1-2\gamma_n}{\gamma_n}\right)$$

This statement captures the intuitive notion that V^1_{NS} and V^2_{NS} should contain the same identities up to differences expected under the γ_n bound. A view is γ_n-*true* if it is γ_n-similar to the true view.

Lemma 4. *Under Restriction 1, the view generated by any truthful receiver set of size n is γ_n-consistent*[6].

[5] γ_n is not the probability that an individual identity is misclassified, but an upper bound on the total fraction misclassified.

[6] This assumes that the false negative bound is negligible. If it is not, a similar notion of γ,σ-consistency,

Proof. Consider two views V^1 and V^2 generated by conforming receiver sets. Each correctly classifies at least $(1-\gamma_n)$ of the non-Sybil identities, so $|V^1_{\mathrm{NS}} \cap V^2_{\mathrm{NS}}| \geq (1-2\gamma_n)|NS|$. Each misclassifies at most γ_n of the non-Sybil identities, so $|V^1_{\mathrm{NS}} \setminus V^2_{\mathrm{NS}}| \leq \gamma_n|NS|$ and similar for $V^2_{\mathrm{NS}} \setminus V^1_{\mathrm{NS}}$. The ratio of these bounds is the result. □

Substituting γ-consistency for pure consistency, Condition 3 still holds with high (albeit different) probability, quantified in Section 5.5. An analogue of Theorem 2 follows.

Theorem 3. *Under Condition 3, the γ_n-consistent view of the largest subset of I permitting such a view is γ_n-true.*

In Section 5.4 we describe an efficient algorithm to identify the largest subset permitting a γ-consistent view and thus the correct (up to errors expected due to signalprint noise) Sybil classification.

5.4 Efficient Implementation of the Selection Policy

Algorithm 1 Choose the receiver sets to consider

Require: i_0 is the identity running the procedure
Require: n is the desired receiver set size
1: $S \leftarrow \emptyset$
2: **for all** $i \in I$ **do**
3: $\quad R \leftarrow \{i_0, i\}$
4: $\quad$ **for** $cnt = 3 \rightarrow n$ **do**
5: $\quad\quad R \leftarrow R \cup \{\text{RandElement}(V_{\mathrm{NS}}(R))\}$
6: $\quad$ **end for**
7: $\quad S \leftarrow S \cup \{R\}$
8: **end for**
9: **return** S ▷ with high probability, S contains a truthful receiver set

Theorem 3 suggests a way to identify a γ_n-true view, but brute-force examination of all $2^{|I|}$ receiver sets is infeasible. Instead, we give an $O(|I|^3)$ approach. The first algorithm

where σ is the false negative bound, can be used. In practice σ is quite small [5, 109], so simple γ_n-consistency is fine.

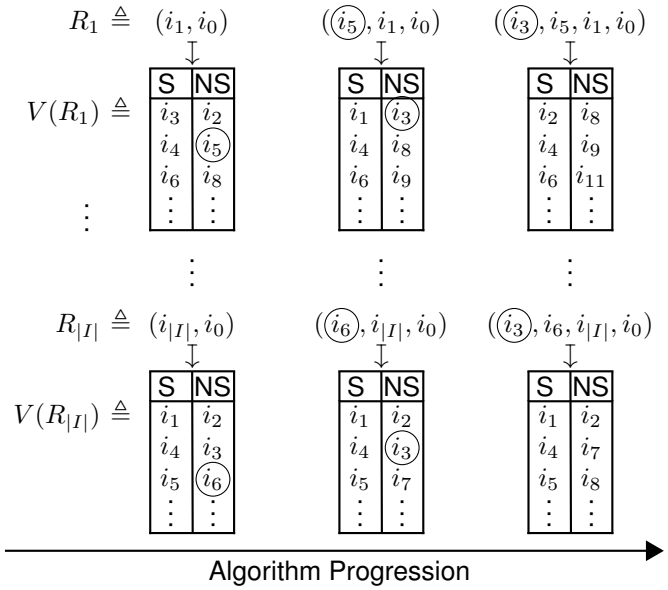


Figure 5.5: Illustration of Algorithm 1. All $|I|$ size-2 receiver sets are increased to size-4 by iteratively adding a random identity from those labeled non-Sybil by the current set. With high probability, at least one of the final sets will contain only conforming identities.

picks $O(|I|)$ receiver sets to consider and the second identifies that permitting the largest γ_n-consistent subset.

5.4.1 Receiver Set Selection

The only requirement for receiver set selection is that at least one of the chosen receiver sets must be truthful. Algorithm 1 selects $|I|$, size-n (we suggest $n = 4$) receiver sets of which at least one is truthful with high probability. As illustrated in Figure 5.5, the algorithm starts with all $|I|$ size-2 receiver sets (lines 2–3) and builds each up to the full size-n by iteratively (line 4) adding a randomly selected identity from those indicated to be conforming at the prior lower dimensionality (line 5). At least $|C|$ of the initial size-2 receiver sets are conforming and after increasing to size-n, at least one is still conforming with high probability (graphed in Figure 5.6):

$$1 - \left(1 - \prod_{m=2}^{n-1} \frac{(1-\gamma_m)\cdot|C| - (m-1)}{|LNS| + (1-\gamma_m)\cdot|C| - (m-1)}\right)^{|C|}$$

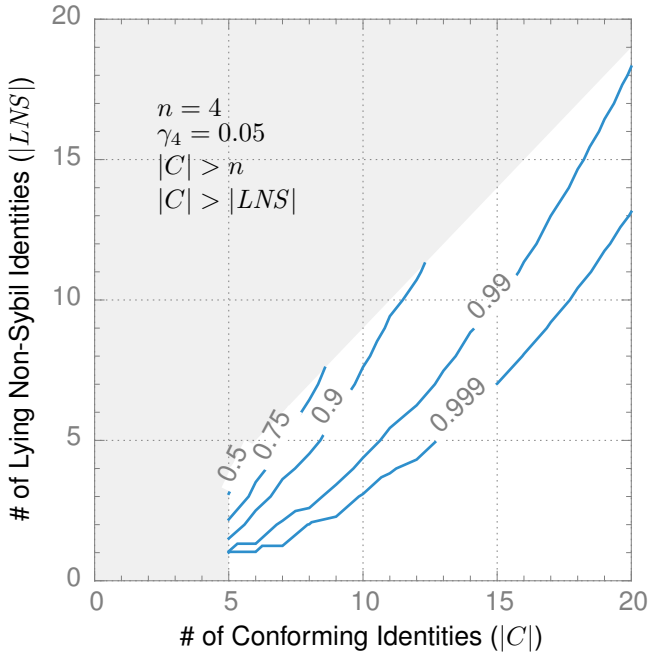


Figure 5.6: Contours of probability that at least one of the receiver sets from Algorithm 1 is conforming[7].

The signalprint threshold for this process is chosen to eliminate (nearly) all false negatives, as having a few false positives does not matter. The complexity of a straightforward implementation is $O(|I|^3)$. Section 5.9 further discusses the runtime.

5.4.2 Finding the Largest γ_n-Consistent View

Given the $|I|$ candidate receiver sets, the next task is identifying the one generating a γ_n-true view, which, pursuant to Theorem 3, is that permitting the largest subset of I to be γ_n-consistent. Checking consistency by examining all $2^{|V_{\text{NS}}|}$ receiver sets is infeasible, so we make several observations leading to the $O(|I|^3)$ Algorithm 2. For each candidate receiver set (line 2), we determine how many identities must be excluded for the view to be γ_n-consistent (lines 3–17). That excluding the fewest is γ_n-true and the desired classification (line 22).

[7]For small $|C|$ and relatively large $|LNS|$ the probability can be increased by building $2 \cdot |I|$ or $3 \cdot |I|$ or more receiver sets instead. We omit further details due to lack of space.

Algorithm 2 Find receiver set permitting the largest γ_n-consistent subset

Require: S is the set of receivers sets generated by Algorithm 1
Require: $V_{\mathrm{NS}}(R)$ for each $R \in \{\text{size-2 receiver sets}\}$ computed by Algorithm 1
Require: s is the initiator running the algorithm

$(C, R_{\text{max}}) \leftarrow (\infty, \text{null})$
for all $R \in S$ **do**
    Compute RSSI ratio for each Sybil set in $V_{\mathrm{S}}(R)$
    $c \leftarrow 0$
    for all $i \in V_{\mathrm{NS}}(R)$ **do**
        $e \leftarrow 0$
        $n \leftarrow$ number of identities whose RSSI ratios reported by i do not match that for R
        if $\frac{|V_{\mathrm{NS}}(R)|+n}{n} < \frac{1-2\gamma_n}{\gamma_n}$ **then**
            $e \leftarrow 1$
        end if
        if $V(R)$ and $V(\{i, s\})$ are not γ_2-similar **then**
            $e \leftarrow 1$
        end if
        if $e = 1$ **then**
            $c \leftarrow c + 1$ ▷ exclude i
        end if
    end for
    if $c < C$ **then**
        $(C, R_{\text{max}}) \leftarrow (c, R)$ ▷ new largest γ-consistent subset found
    end if
end for
return R_{max}

The crux of the algorithm is lines 3–17, which use the following observations to efficiently determine which identities must be excluded.

1. Adding an identity to a receiver set can change the view in one direction only—an identity can go from Sybil to non-Sybil, but not vice versa—because uncorrelated RSSI vectors cannot become correlated by increasing the dimension[8].

2. For identities a and b, $R \cup \{a\} \mapsto V(R)$ and $R \cup \{b\} \mapsto V(R)$ implies $R \cup \{a, b\} \mapsto V(R)$ because a and b must have the same RSSI ratios for the Sybils as R.

From these observations, we determine the excluded identities by computing, for each identity in $V_{\mathrm{S}}(R)$, the RSSI ratio with an arbitrary sibling (line 3) and comparing against those reported by potential non-Sybils in $V_{\mathrm{NS}}(R)$ (line 7). If the number not matching is too large (line 8), the view is not γ_n-consistent and the identity is excluded (line 15). Further, if the receiver set consisting of just the supposedly non-Sybil identity and the initiator is not γ_2-similar to R (line 11).

5.4.3 Runtime in the Absence of Liars

In a typical situation with no liars, the algorithm can detect the Sybils in $O(|I|^2)$ time. Since all identities are truthful, any chosen receiver set will be γ_n-consistent with no exclusions—clearly the largest possible—and thus the other $|I| - 1$ also-truthful receiver sets need not be checked. With lying attackers present, the overall runtime is $O(|I|^3)$, as each algorithm takes $O(|I|^3)$ time.

5.5 Probability that Critical Conditions Hold

Our Sybil classification method depends on Condition 3 holding, i.e., all consistent views must correctly classify at least $|LNS| + 1$ conforming identities. In this section, we quantify

[8]This is not true for low dimension receiver sets affected by noise, but is for the size-$(n > 4)$ sets considered here.

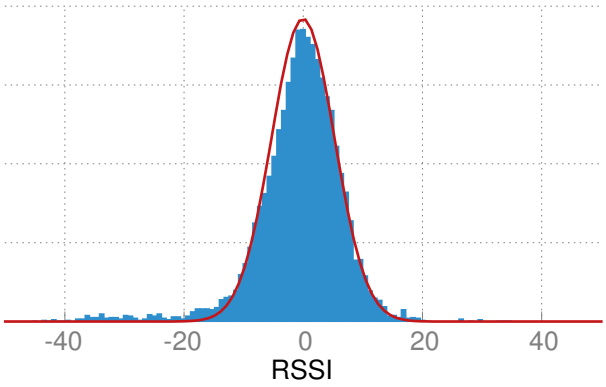


Figure 5.7: Distribution of RSSI variations in real-world deployment.

the probability that Condition 3 holds in the presence of the optimal attacker strategy.

5.5.1 RSSI Unpredictability

The probability that Condition 3 holds is directly tied to the unpredictability of RSSIs, because collapsing identities requires knowing the RSSI observations at the initiator, as explained in Subsection 5.3.1. Accurate guessing is difficult because the wireless channel varies significantly with small displacements in location and orientation (*spatial variation*) and environmental changes over time (*temporal variation*) [107, 119]. Pre-characterization could account for spatial variation, but would be prohibitively expensive at the needed spatial and orientation granularity (6.25 cm [120] and 3° for our test devices).

We empirically determined the RSSI variation for human-carried smartphones by deploying experimental phones to eleven graduate students in two adjacent offices and measuring fixed-distance, pairwise RSSIs for fifteen hours. The observed distribution of deviations[9], shown in Figure 5.7, is roughly normal with a standard deviation of 7.3 dBm, in line with other real-world measurements for spatial and orientation variations (4–12 dBm and 5.3 dBm [107]). We use this distribution to model the attacker uncertainty of RSSIs.

[9]For each pair of transceivers, we subtracted the mean of all their measurements to get the deviations and aggregated all the pairwise deviations.

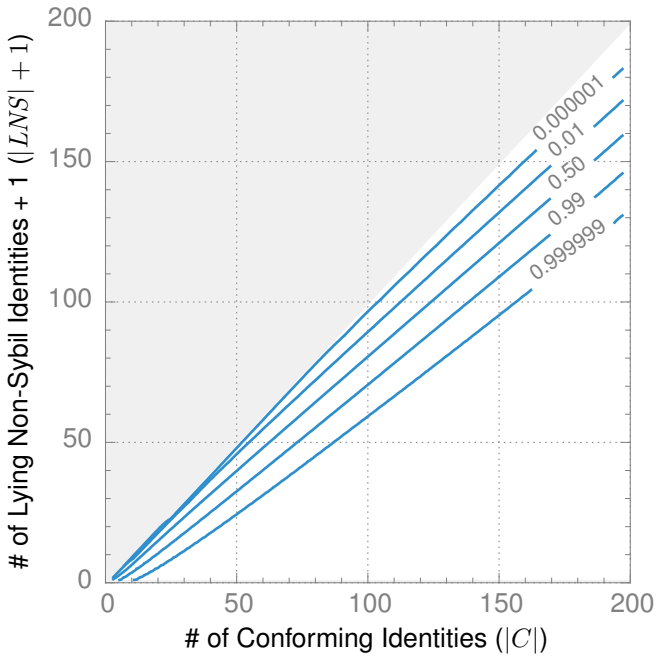


Figure 5.8: Contours of a lower bound on the probability that Condition 3 holds under an optimal attacker strategy with the attacker's knowledge of RSSIs modeled as a normal distribution with standard deviation 7.3 dBm.

5.5.2 Optimal Attacker Strategy

To break Condition 3, an attacker must generate a consistent view that collapses at least $|C| - |LNS|$ conforming identities. We give three observations about the optimal attacker strategy for this goal.

1. Collapsing $|C| - |LNS|$ identities is easiest with larger $|LNS|$. Thus, the optimal attacker uses only one physical node to claim Sybils—the others just lie.

2. For a particular false view to be consistent, all supposedly non-Sybil identities must indict the same identities, e.g., have the same RSSI guesses for the collapsed conforming identities. The optimal attacker must divide its (mostly Sybil) identities into groups, each using a different set of guesses.

3. More groups increases the probability of success, but decreases the number of Sybils actually accepted, as each group is smaller.

We assume the optimal attacker wishes to maximize the probability of success and thus uses minimum-sized groups (three identities, for size-4 signalprints).

For each group, the attacker must guess RSSI values for the conforming identities with the goal of collapsing at least $s \triangleq |C| - |LNS|$ of them. There are $\binom{|C|}{s}$ such sets and the optimal attacker guesses values that maximize the probability of at least one (across all groups) being correct. The first group is easy; the $|C|$ guesses are simply the most likely values, i.e., the expected values for the conforming identities' RSSIs, under the uncertainty distribution.

For the next (and subsequent) groups, the optimal attacker should pick the next most likely RSSI values for each of the $\binom{|C|}{s}$ sets. However, the sets share elements (only $|C|$ RSSIs are actually guessed), so the attacker must determine the most probable sets that are compatible. This is a non-trivial problem, so for our analysis, we assume that all sets are compatible (e.g., one group can guess −78 dBm and −49 dBm RSSIs for nodes a and b, but −82 dBm and −54 dBm RSSIs for nodes a and c). This is clearly impossible, but leads to a conservative lower bound on the probability that the attacker fails—a feasible, optimal strategy is less likely to succeed.

Figure 5.8 shows contours of this lower bound on the probability that Condition 3 holds as a function of $|C|$ and $|LNS|$. When the conforming nodes outnumber the attacker nodes by at least 1.5× —the expected case in real networks—the condition holds with very high probability. In practice, it will hold with even higher probability, as this is a lower bound.

5.6 Detecting Mobile Attackers

A mobile attacker can defeat signalprint comparison by changing locations or orientations between transmissions to associate distinct signalprints with each Sybil identity. Fortunately, in the networks we consider, most conforming nodes (e.g., human-carried smartphones and laptops) are stationary over most short time-spans (1–2 min), due to human mobility habits.

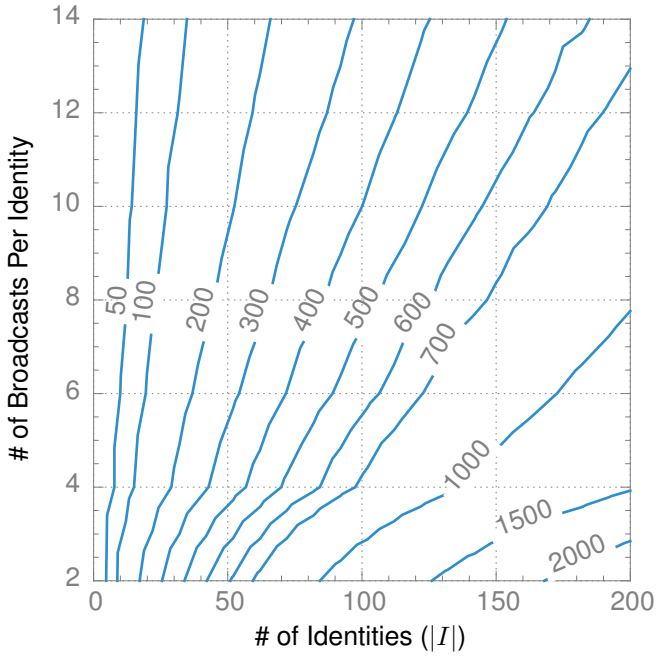


Figure 5.9: Contours showing the response time (in ms, 99$^{\text{th}}$ percentile) to precisely switch between two positions required to defeat the challenge-response moving node detection.

Thus, multiple transmissions should have the same signalprint, as noted, but not pursued, by Xiao et al. [5]. From this observation, we develop a protocol to detect moving attackers.

Again, the lack of trusted observations is troublesome. Instead of comparing signalprints, we compare the initiator's observations: all transmissions from a conforming node should have the same RSSI. As shown in Section 5.8, this simple criterion yields acceptable detection.

The protocol collection phase (Figure 5.2a) is extended to request multiple probe packets (e.g., 14) from each identity in a pseudo-random order (see Subsection 5.7.1). During the classification phase (Figure 5.2c) each participant rejects all identities with large RSSI variation across its transmissions (specifically, a standard deviation larger than 2.5 dBm). In essence, an attacker is challenged to quickly and precisely switch between the multiple positions associated with its Sybil identities (6.25 cm location precision according to coherence length theory [120] and 3° orientation precision according to our measurements).

Figure 5.9 plots the required response time for an attacker to pass the challenge. Given

human reaction times [121], reliably mounting such an attack would require specialized hardware—precise electromechanical control or beam steering antenna arrays—that is outside our attack model and substantially more expensive to deploy than compromised commodity devices.

5.7 The Mason Test

This section presents the full Mason test protocol, implementing the concepts introduced in the previous sections. Those results impose four main requirements on the protocol.

1. Conforming neighbors must be able to participate. That is, selective jamming of conforming identities must be detectable.

2. Probe packets must be transmitted in a pseudo-random order. Further, each participant must be able to verify that no group of identities controlled the order.

3. Moving identities are rejected, so to save energy and time, conforming nodes that are moving when the protocol begins should not participate.

4. Attackers must not know the RSSI observations of conforming identities when computing lies.

We assume a known upper bound on the number of conforming neighbors, i.e., those within the one-hop transmission range. In most applications, a bound in the hundreds (we use 400 in our experiments) will be acceptable. If more identities attempt to participate, the protocol aborts and no classification is made. This appears to open a denial-of-service attack. However, we do not attempt to prevent, instead only detect, DOS attacks, because one such attack—simply jamming the wireless channel—is unpreventable (with commodity hardware). See Section 5.9 for more discussion.

The rest of this section describes the two components of the protocol: collection of RSSI observations and Sybil classification. We assume one identity, the initiator, starts the

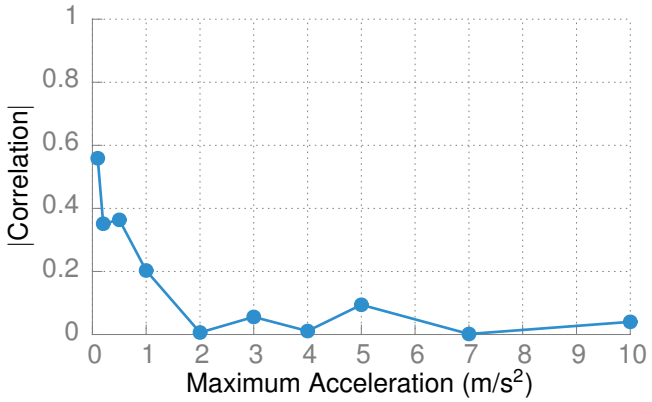


Figure 5.10: RSSI correlation as a function of the maximum device acceleration between observations.

protocol and leads the collection, but all identities still individually and safely perform Sybil classification.

5.7.1 Collection of RSSI Observations

Phase I: Identity Collection. The first phase gathers participating neighbors, ensuring that no conforming identities are jammed by attackers. The initiator sends a REQUEST message stating its identity, e.g., a public key. All stationary neighbors respond with their identities via HELLO-I messages, ACKed by the initiator. Unacknowledged HELLO-Is are re-transmitted. The process terminates when the channel is idle—indicating all HELLO-I's were received and ACKed. If the channel does not go idle before a timeout (e.g., 15 seconds), an attacker may be selectively jamming some HELLO-Is and the protocol aborts. The protocol also aborts if too many identities (e.g, 400) join.

Phase II: Randomized Broadcast Request: The second phase is the challenge-response protocol to collect RSSI observations for motion detection and Sybil classification. First, each identity contributes a random value; all are hashed together to produce a seed to generate the random sequence of broadcast requests issued by the initiator. Specifically, it sends a TRANSMIT message to the next participant in the random sequence, who must

quickly broadcast a signed HELLO-II, e.g., within 10 ms in our implementation[10]. Each participant records the RSSIs of the HELLO-II messages it hears. Some identities will not hear each other; this is acceptable because the initiator needs observations from only three other conforming identities. $|I| \times s$ requests are issued, where s is large enough to ensure a short minimum duration between consecutive requests for any two pairs of nodes, e.g., 14 in our tests. An identity that fails to respond in time might be an attacker attempting to change positions and is rejected.

Phase III: RSSI Observations Report. In the third phase, the RSSI observations are shared. First, each identity broadcasts a hash of its observations. Then the actual values are shared. Those not matching the respective hash are rejected, preventing attackers from using the reported values to fabricate plausible observations. The same mechanism from Phase 1 is used to detect selective jamming.

5.7.2 Sybil Classification

Each participant performs Sybil classification individually. First, the identity verifies that its observations were not potentially predictable from those reported in prior rounds, possibly violating Condition 3. Correlation in RSSI values between observations decreases with motion between observations, as shown by our experiments (Figure 5.10). Thus, a node only performs Sybil classification if it has strong evidence the current observations are uncorrelated with prior ones[11], i.e., it has observed an acceleration of at least $2\,\mathrm{m\,s^{-2}}$.

Classification is a simple application of the methods of Section 5.6 and Section 5.4. Each identity with an RSSI variance across its multiple broadcasts higher than a threshold is rejected. Then, Algorithm 1 and Algorithm 2 are used to identify a γ-true Sybil classification over the remaining, stationary identities.

[10]These packets can be signed ahead of time and cached—signatures do not need to be computed in the 10 ms interval.

[11]Note that although we did not encounter this case in our experiments, it is conceivable that some devices will return to the same location and orientation after motion.

Table 5.2: Thresholds for Signalprint Comparison and Motion Filtering

Name		**Threshold** (dBm)
Signalprint Distance	dimension-2	0.85
	dimension-3	3.6
	dimension-4	1.2
RSSI Standard Deviation		2.5

5.8 Prototype and Evaluation

We implemented the Mason test as a Linux kernel module and tested its performance on HTC Magic Android smartphones in various operating environments. It sits directly above the 802.11 link layer, responding to requests in interrupt context, to minimize response latency for the REQUEST–HELLO-II sequence (12 ms roundtrip time on our hardware). For rapid prototyping, the classification algorithms are implemented in Python.

Wireless channels are environment-dependent, so we evaluated the Mason test in four different environments.

Office I Eleven participants in two adjacent offices for fifteen hours.

Office II Eleven participants in two adjacent offices in a different building for one hour, to determine whether performance varies across similar, but non-identical environments.

Cafeteria Eleven participants in a crowded cafeteria during lunch. This was a rapidly-changing wireless environment due to frequent motion of the cafeteria patrons.

Outdoor Eleven participants meeting in a cold, open, grassy courtyard for one hour, capturing the outdoor environment. Participants moved frequently to stay warm.

In each environment, we conducted multiple trials with one Sybil attacker[12] generating 4, 20, 40, and 160 Sybil identities. The gathered traces were split into testing and training sets.

[12] As discussed in Section 5.3 and Section 5.5, additional physical nodes are best used as lying, non-Sybils.

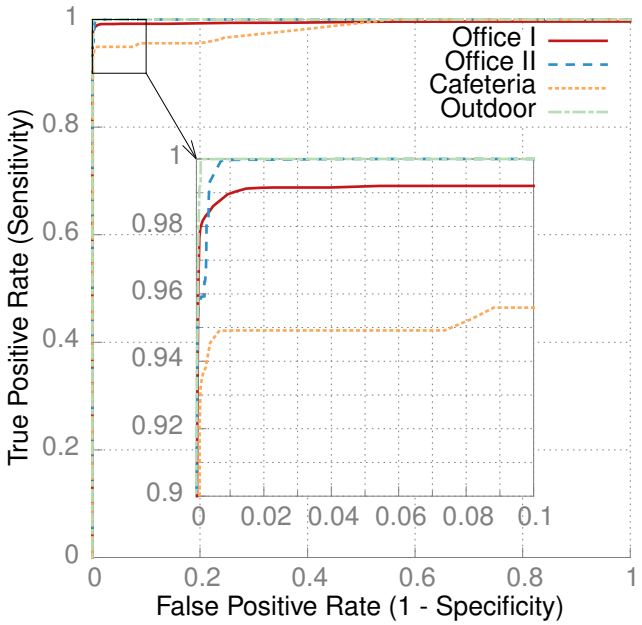


Figure 5.11: ROC curve showing the classification performance of signalprint comparison in different environments for varying distance thresholds. Only identities that passed the motion filter are considered. The knees of the curves all correspond to the same thresholds, suggesting that the same value can be used in all locations.

5.8.1 Selection and Robustness of Thresholds

The training data were used to determine good motion filter and signalprint distance thresholds, shown in Table 5.2.

The motion filter threshold was chosen such that at least 95% of the conforming participants (averaged over all training rounds) in the low-motion Office I environment would pass. This policy ensures that conforming smartphones, which are usually left mostly

Table 5.3: Classification Performance

Environment	Sensitivity (%)	Specificity (%)
Office I	99.6	96.5
Office II	100.0	87.7
Cafeteria	91.4	86.6
Outdoor	95.9	61.1

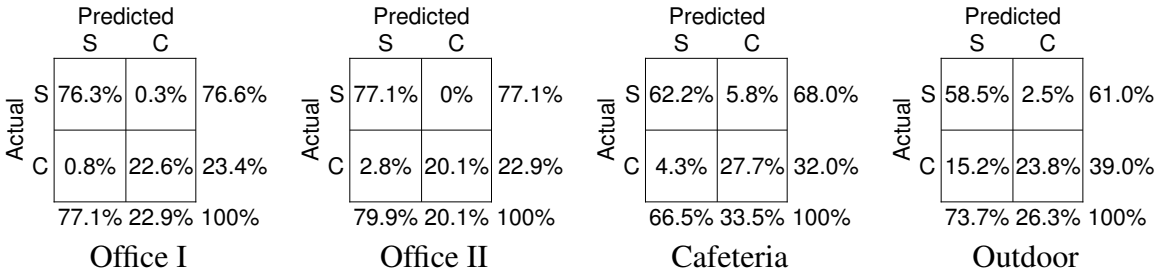


Figure 5.12: Confusion matrices detailing the classifier performance in the four environments tested. *S* means Sybil and *C* means conforming. Multiple tests were conducted in each environment, so mean percentages are shown instead of absolute counts.

stationary, e.g., on desks, in purses, or in the pockets of seated people, will usually pass the test. Devices exhibiting more motion (i.e., a standard deviation of RSSIs at the initiator larger than 2.5 dBm)—as would be expected from an attacker trying to defeat signalprint detection—will be rejected.

The signalprint distance thresholds were chosen by evaluating the signalprint classification performance at various possible values. Figure 5.11 shows the ROC curves for size-4 receiver sets (a "positive" is an identity classified as Sybil). Note that the true positive and false positive rates consider only identities that passed the motion filter, in order to isolate the effects of the signalprint distance threshold. The curves show that a good threshold has performance in line with prior work [5, 109], as expected.

In all environments, the knees of the curve correspond to the same thresholds, suggesting that these values can be used in general, across environments. A possible explanation is that despite environment differences, the signalprint distance distributions for stationary Sybil siblings are identical. All results in this paper use these uniform thresholds, show in Table 5.2.

5.8.2 Classification Performance

The performance of the full Mason test—motion filtering and signalprint comparison—is detailed by the confusion matrices in Figure 5.12. Many tests were conducted in each

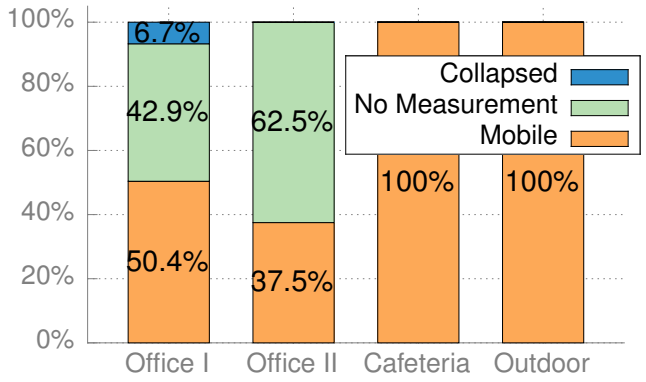


Figure 5.13: Relative frequencies of the three causes of false positives.

environment, so average percentages are shown instead of absolute counts. To evaluate the performance, we consider two standard classification metrics derived from these matrices, *sensitivity* (percentage of Sybil identities correctly identified) and *specificity* (percentage of conforming identities correctly identified).

Note that 100% sensitivity is not necessary. Most protocols that would use Mason require a majority of the participants to be conforming. The total identities are limited (e.g., 400) so rejecting most of the Sybils and accepting most of the conforming identities is sufficient to meet this requirement.

Table 5.3 shows the performance for all four environments. The test performs best in the low-motion indoor environments, with over 99.5% sensitivity and over 85% specificity. The sensitivity in the cafeteria environment is just 91.4%, likely due to the rapid and frequent changes in the wireless environment resulting from the movement of cafeteria patrons. In the outdoor environment, with participants moving, the specificity is 61.1% because some conforming identities are rejected by the motion filter[13].

An identity is classified as Sybil for three reasons: if it has similar signalprints with another, the initiator has too few RSSI reports to form a signalprint, or it fails the motion filter. Figure 5.13 shows the relative prevalence of these causes of the false positives. Interestingly,

[13]These moving participants normally do not participate. Including them here makes clears that they are not identified as conforming.

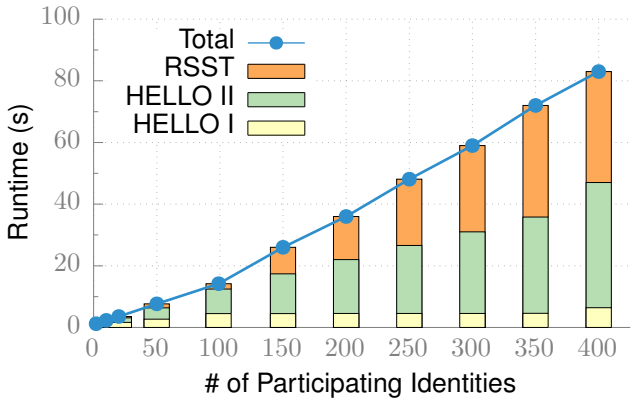


Figure 5.14: Runtime overhead in seconds of the collection phase as a function of the number of participating identities. The stacked bars partition the cost among the participant collection (HELLO I), RSSI measurement (HELLO II), and RSSI observation exchange (RSST) steps.

the first cause—collapsing—is rare, occurring only in the first office environment. Not surprisingly, missing RSSI reports are an issue only in the environments with significant obstructions, the indoor offices, accounting for about half of these false positives. In the open cafeteria and outdoor environments, all false positives are due to participant motion.

5.8.3 Overhead Evaluation

Figure 5.14 and Figure 5.15 show the runtime and energy overhead for the Mason test collection phase, with the stacked bars separating the costs by sub-phase. For a protocol not run frequently (once every hour is often sufficient), the runtimes of 10–90 seconds are acceptable. Likewise, energy consumption is acceptable, with the extreme 18 J consumption for 400 identities representing 0.01% of the 17.500 J capacity of a typical smartphone batteries.

Figure 5.16 show the classification phase overheads for 2–100 identities. At a small fraction of the collection costs, these are also acceptable. For more than 100 participants, costs become excessive due to the $O(n^3)$ scaling behavior[14]. Limiting participation to 100 identities may be necessary for energy-constrained devices, but will generally not reduce

[14] A native-C implementation might scale to 300–400 identities.

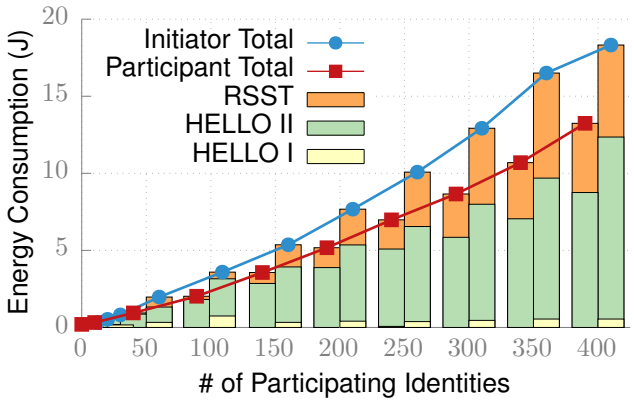


Figure 5.15: Energy consumption in joules of the collection phase as a function of the number of participating identities. The stacked bars partition the cost among the participant collection (HELLO I), RSSI measurement (HELLO II), and RSSI observation exchange (RSST) steps.

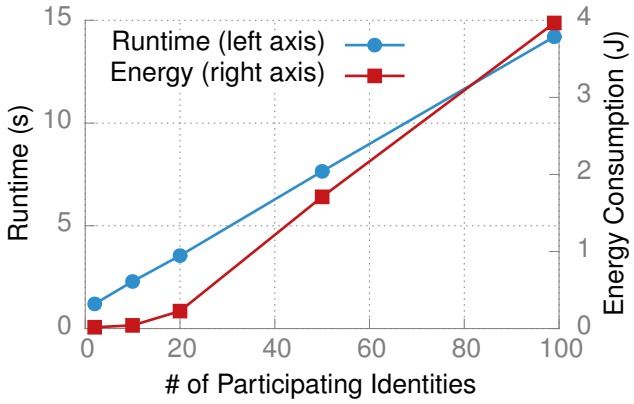


Figure 5.16: Runtime and energy consumption of the classification phase.

performance because having 100 non-Sybil, one-hop neighbors is rare.

The periodic accelerometer sampling used to measure motion between Mason test rounds consumes 5.2% of battery capacity over a typical 18 h work day.

5.9 Discussion

Sybil classification from untrusted observations is difficult and the Mason test is not a silver bullet. Not requiring trusted observations is a significant improvement, but the test's limitations must be carefully considered before deployment. As with any system intended

for real-world use, some decisions try to balance system complexity and potential security weaknesses. In this section, we discuss these trade-offs, limitations, and related concerns.

High Computation Time: The collection phase time is governed by the 802.11b-induced 12 ms per packet latency and the classification runtime grows quickly, $O(|I|^3)$. Although typically fast (e.g., $<$5 s for 5–10 nodes), the Mason test is slower in high density areas (e.g., 40 s for 100 nodes). However, it should be run infrequently, e.g., once or twice per hour. Topologies change slowly (most people change locations infrequently) and many protocols requiring Sybil resistance can handle the lag—they need only know a subset of the current non-Sybil neighbors.

Easy Denial-of-Service Attack: An attacker can force the protocol to abort by creating many identities or jamming transmissions from the conforming identities. We cannot on commodity 802.11 devices solve another denial-of-service attack—simply jamming the channel—so defending against these more-complicated variants is ultimately useless. Most locations will at most times be free of such attackers—the Mason test provides a way to verify this condition, reject any Sybils, and let other protocols operate knowing they are Sybil-free.

Requires Several Conforming Neighbors: The Mason test requires true RSSI observations from some neighbors (i.e., 3) and is easily defeated otherwise. Although beyond the page limits of this paper, protocols incorporating the Mason test can mitigate this risk by (a) a priori estimation of the distribution of the number of conforming neighbors and (b) careful composition of results from multiple rounds to bound the failure probability.

Limit On Total Identities: This limit (e.g., 400) is unfortunately necessary to detect when conforming nodes are being selectively jammed while keeping the test duration short enough that most conforming nodes remain stationary. We believe that most wireless networks have typical node degrees well below 400 anyway.

Messages Must Be Signed: Packets sent during the collection phase are signed, which can be very slow with public key schemes. However, this is easily mitigated by (a) pre-signing

the packets to keep the delay off the critical path or (b) using faster secret-key-based schemes. Details are again omitted due to page constraints.

Pre-Characterization Reveals RSSIs: An attacker could theoretically improve its collapsing probability by pre-characterizing the wireless environment. We believe such attacks are impractical because the required spatial granularity is about 6.25 cm, the device orientation is still unknown, and environmental changes (e.g., people moving) reduces the usefulness of prior characterization.

Prior Rounds Reveal RSSI Information: The protocol defends against this. Conforming nodes do not perform classification when their RSSI observations are correlated with those from the prior rounds (see Subsection 5.7.2).

High False Positive Rates: With the motion filter, the false positive rate can be high, e.g., 20% of conforming identities rejected in some environments. We believe this is acceptable because most protocols requiring Sybil resistance need only a subset of honest identities. For example, if for reliability some data is to be spread among multiple neighbors, it is acceptable to spread it among a subset chosen from 80%, rather than all, of the non-Sybils.

5.10 Conclusion

We have described a method to use signalprints to detect Sybil attacks in open ad hoc and delay-tolerant networks without requiring trust in any other node or authority. We use the inherent difficulty of predicting RSSIs to separate true and false RSSI observations reported by one-hop neighbors. Attackers using motion to defeat the signalprint technique are detected by requiring low-latency retransmissions from the same position.

The Mason test was implemented on HTC Magic smartphones and tested with human participants in three environments. It eliminates 99.6%–100% of Sybil identities in office environments, 91% in a crowded high-motion cafeteria, and 96% in a high-motion open outdoor environment. It accepts 88%–97% of conforming identities in the office environments,

87% in the cafeteria, and 61% in the outdoor environment. The vast majority of rejected conforming identities are removed due to motion.

CHAPTER 6

Characterization of Microblogging User Behavior and the Retweet Graph

6.1 Introduction

Quantitative modeling of Twitter usage is important both for understanding human communication patterns and optimizing the performance of other microblogging-esque communication platforms. However, prior analysis is focused on the social graph [122–126] or on individual information cascades that represent a small fraction of all tweets [127–131]. Descriptions of basic behaviors are missing from the literature. For example, the qualitative distributions of the number of *followers* and *friends* is available [122], but not the distribution of tweet rates. Common factors of tweets that are heavily retweeted are known [127], but propensity of users to retweet, i.e., distribution of retweet rates, is not. We begin to fill these gaps by considering user behavior *as a whole*, providing quantitative descriptions of the distributions of lifetime tweets, tweet rates, and inter-tweet times.

We are motivated by increasing interest in decentralized microblogging systems designed to protect user privacy and resist censorship. FETHER [69], Cuckoo [70], and Litter [71] reduce dependence on a single provider, while Shout [132] and Twister [133] are explicitly designed to avoid censorship and reprisal by government agencies. Designing a decentralized system capable of handling the message rates and volumes of Twitter is already a significant challenge and is nearly impossible without a good understanding of those usage patterns.

Given the complexity of these systems, understanding of the trade-offs in the performance and cost metrics—throughput, latency, energy consumption—is obtained through simulations, but such simulations are only as accurate as the data and models driving them. Consider fair allocation of network resources—fairness looks very different when the *expected* distribution of tweets is a power law and not uniform. Or, consider measuring delivery latencies, with messages queuing at intermediate nodes, a metric dependent on the (non-Poisson) arrival process, i.e., the inter-tweet duration distribution. Quantitative models of these basic behaviors are needed.

The underlying human behaviors should extend across communication platforms—tweet rates should mirror call rates in the telephone network and total lifetime tweets should mirror total lifetime contributions to Wikipedia or YouTube—suggesting that models of those behaviors [134–136] be used in proxy for microblogging design. However, our analysis of the Twitter data shows differing behavior, indicating possible faults in several of these models. Our results for Twitter should enable future work to identify or refine further commonalities in human communication.

Tweets generally travel via the explicit social *followers* graph [122], which has been well-studied. Surprisingly, the *retweet* graph, in which a directed edge connects two users if the source has retweeted the destination, has received almost no attention. This *implicit* graph may be actually more relevant to information propagation in decentralized systems. A throughput-limited system needs some way of prioritizing messages. People are usually more selective in what they say than to whom they listen, so the retweet graph may better encode true interest and trust relationships among users. For example, Shout[1] does not support friend/follower relationships, so the retweet graph is the only available social graph. We conduct the first study of the retweet graph obtained from a 4-month sample of 10% of all tweets and compare it to the social followers graph.

[1] Shout [132] is decentralized, geographic microblogging system in which messages are broadcast to users within radio range of the sender. Other users may re-shout the message, extending its reach, but the protocol does not directly support multi-hop delivery.

These results have wide applicability. The quantification of communication behaviors and the social graph, beyond allowing direct comparison with other already-characterized platforms, enables the development of generative models explaining the underlying processes. In a more direct view, knowing the number of tweets, tweet rates, and inter-tweet times are sufficient for simulating and optimizing microblogging platform performance and the confirmation that the retweet graph is scale-free and small-word enables the generation of random retweet graphs for empirical evaluation. We focus on two such applications, the design of distributed microblogging systems and the detection of spammers using connectivity in the retweet graph.

We have the following findings.

- The distribution of lifetime tweets is discrete Weibull (type-II), generalizing a power law form shown by Wilkinson for other online communities [134]. We conjecture that the Weibull shape parameter reflects the average amount of (positive or negative) feedback available to contributors. (Section 6.3)

- The distribution of tweet (and retweet) rates is asymptotically power law, but exhibits a lognormal cutoff over finite-duration samples. Thus, high tweet rates are much more rare in practice than the asymptotic distribution would suggest. We also discount a double Pareto lognormal (DPLN) explanation previously advanced in the context of call rates [135]. (Section 6.4)

- The distribution of inter-tweet durations is power law with exponential cutoff, mirroring that of telephone calls [136]. (Section 6.5)

- The retweet graph is small-world and (roughly) scale-free, like the social followers graph, but less disassortative and more highly clustered. It is more similar than the followers graph to real-world social networks, consistent with better reflection of real-world relationships and trust. (Section 6.6)

In Section 6.7, we discuss the implications of these results for decentralized microblogging architectures and in Section 6.8 we consider using the structure of the retweet graph for spammer detection.

6.2 Datasets

The Twitter API rate limits and terms of service prevent collection and sharing of a single complete tweet dataset suitable for all our queries [137]. Our largest and most recent dataset—10% of all tweets sent between June and September 2012—is the focus of our analysis, but we supplement with sets from other researchers as necessary. This section summarizes these datasets and describes our main procedure for inferring population statistics from the 10% sample.

6.2.1 2009 Social Graph

Kwak et al.'s 2009 crawl [122] remains the largest and most complete public snapshot of the Twitter social followers graph, covering 41.7 million users and 1.47 billion relations. The data is dated, but still the best available. Repeating this crawl is infeasible under current rate limits and feasible sampling strategies (e.g., snowball-sampling [138]) lead to results that are difficult to interpret [139]. We use this social graph snapshot for all comparisons with the retweet graph.

6.2.2 Lifetime Contribution Dataset

No tweet dataset is complete enough to compute lifetime contributions, the number of tweets sent before quitting Twitter, but the Twitter API exposes (subject to rate limits) the necessary information. We collected account age, date of last tweet, and total tweet count (as of June

Table 6.1: 10% Sample (Gardenhose) Dataset

	10% Sample	**Actual Value**†
# of Tweets	4 097 787 713	41 256 584 408
# of Retweets	953 457 874	9 664 691 519
# of Tweeters	104 083 457	166 335 390
# of Retweeters	51 319 979	84 278 086
# of Retweetees	38 975 108	69 224 526

† Estimated using the described EM procedure.

2013) for 1 318 683 users selected uniformly randomly from the 2009 social graph set[2]. 525 779 of these users were inactive,[3] i.e., had not tweeted in the prior six months [134]. Their ages and tweet counts form the lifetime contribution set used in Section 6.3.

6.2.3 SNAP Tweet Dataset

Computing inter-tweet intervals requires consecutive tweets—a random sample is insufficient[4]. For our inter-tweet distribution analysis in Section 6.5, we use a collection of 467 million tweets gathered by the SNAP team in 2009 [140]. The full dataset is no longer publicly available per request from Twitter, but the authors kindly shared the inter-tweet metadata.

6.2.4 10% Sample (Gardenhose) Dataset

Our primary dataset is a uniform random 10% sample[5] of all tweets (the "gardenhose" stream) sent in the four month period spanning June through September 2012. Table 6.1 shows the scope of the dataset, using the following definitions. A *tweeter* is a user that sends a *tweet*, an original message. A *retweeter* is a user that sends a *retweet*, forwarding a

[2]The 2009 social graph dataset is the closest to a uniform random sample of Twitter users we could find. More recent sets are biased towards users that tweet more often.

[3]The creation dates of protected tweets are hidden, so all users with protected tweets were excluded.

[4]A random sample would be sufficient if the process were Poisson, but it is not.

[5]More precisely, each tweet is included in this sample with 10% probability.

previous tweet. A *retweetee* is a user whose tweet was *retweeted*.

Retweets were identified using both Twitter-provided metadata and analysis of the message contents for retweet syntax, e.g., "RT@". Retweeting was not an official feature in Twitter's early years, but instead developed organically. A variety of syntaxes appeared (e.g., `RT@username`, `retweeting username`, and `via username`) and are still used today. We detect such retweets using the following (Java) regular expression.

```
Pattern.compile(
 "(?:^|[\\W])(?:rt|retweet(?:ing)?|via)" +
 "\\s*:?\\s*@\\s*([a-zA-Z0-9_]{1,20})"   +
 "(?:\$|\\W)"
)
```

In 2009, Twitter officially[6] added support for retweeting to their backend schema and the user interface. These retweets are identified by the Twitter API.

The sampled data poses a challenge for drawing quantitative conclusions about user behavior and the structure of the retweet graph. For many of the distributions we wish to quantify, the sample is biased towards users that tweeted more frequently. In fact, most users with fewer than ten tweets will not appear at all. Much prior work in the social network and graph analysis literature has focused on qualitatively characterizing the errors introduced by subsampling, motivated by quicker analysis [139, 141]. We instead develop an approach to accurately estimate quantitative population statistics from the 10% random sample.

6.2.5 Estimating Population Distributions from the 10% Sample Dataset

For simplicity, we describe the method for a concrete problem: determining the distribution of tweets per user during the four month window. The method is trivially adapted to a variety of such problems, including multivariate joint distributions as in Subsection 6.6.3. Similar

[6] https://blog.twitter.com/2009/project-retweet-phase-one

approaches are used in other fields [142]. We wish to determine the number of users, f_i, with $i \in \mathbb{N}^+$ tweets given the number of users, g_j, with $j \in \mathbb{N}^+$ tweets observed in the sample. g_j includes some users from each $f_{i \geq j}$, with the binomial distribution $B_{0.1}(i, j)$ describing how the users in f_i are partitioned among the various $g_{j \leq i}$. Intuitively, a good estimate $\hat{f}$ is that which maximizes the probability of the observation g, i.e., standard maximum likelihood estimation.

The corresponding likelihood function is not analytically tractable, so we employ an expectation maximization algorithm [143, 144] to compute the estimate $\hat{f}$, summarized here (see Section 6.9 for details). Let ϕ_i be the probability that a user sends i tweets conditional on at least one of them being observed and $c_{i,j}$ be the probability a user with i tweets has j of them observed conditional on $j \geq 1$ (i.e., the binomial probability conditional on at least one success). The log-likelihood function to maximize is

$$\mathcal{L}(\phi | f, g) = \sum_{1 \leq j \leq i} f_{i,j} \log \left(\phi_i c_{i,j} \right), \tag{6.1}$$

where ϕ are the parameters to estimate and f and g are the hidden and observed variables, respectively. We compute the parameter estimate by iteratively selecting a new estimate ϕ^{k+1} that maximizes the *expected* likelihood under the previous estimate ϕ^k, i.e.,

$$\phi^{k+1} \triangleq \arg \max_{\phi} \mathcal{Q} \left(\phi, \phi^k \right), \tag{6.2}$$

where

$$\mathcal{Q} \left(\phi, \phi^k \right) \triangleq \mathsf{E}_{f|g,\phi^k} \left[\mathcal{L}(\phi | f, g) \right]. \tag{6.3}$$

This process is known to converge [145]. Letting $\gamma \triangleq \sum_{1 \leq j} g_j$ be the total number of observed users, Equation 6.2 can be solved using Lagrangian multipliers to yield

$$\phi_i^{k+1} = \frac{1}{\gamma} \mathsf{E}_{\phi^k} \left[f_i | g \right] \tag{6.4}$$

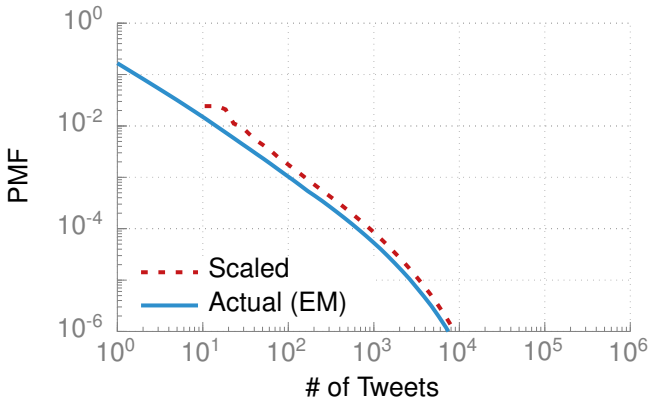


Figure 6.1: Distribution of tweets per user for the scaled sample (j observed tweets maps to $10j$ sent tweets) and the underlying population as estimated by the EM algorithm. The differences (particularly for the range 1–100) illustrate the importance of recovering the actual distribution via, for example, our EM algorithm.

and the hidden original frequencies recovered from the final estimate $\hat{\phi}$ as

$$\hat{f}_i = \gamma \hat{\phi}_i \frac{1}{1 - B_{0.1}(i, 0)}. \tag{6.5}$$

Figure 6.1 shows the result using the distribution of tweets sent during our four-month collection window as an example. The correct distribution computed via the EM algorithm is substantially different, particularly in the lower decades, from the uncorrected or scaled (i.e., assuming that observing j tweets implies $10j$ were sent) distributions.

6.3 Distribution of Lifetime Tweets

Strong regularities in participation behavior have been observed across many online peer production systems, suggesting a common underlying dynamic. Wilkinson found that for Bugzilla, Essembly, Wikipedia, and Digg, the probability that a user makes no further contributions is inversely proportional to the number of contributions already made, suggesting a notion of *participation momentum* [134]. Huberman et al. observed the same in YouTube [146]. We look for a similar effect in Twitter.

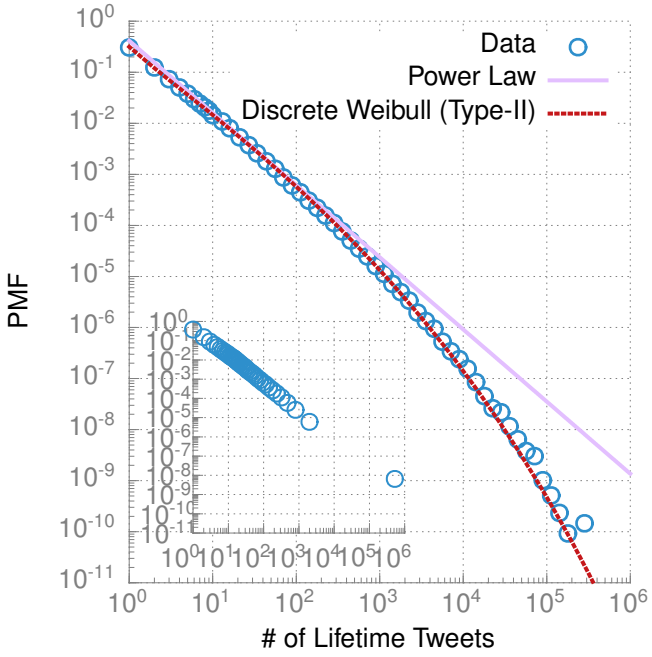


Figure 6.2: Distribution of total lifetime tweets. Distribution parameters (Table 6.3) were obtained by maximum likelihood estimation. In the inset, equal-count binning obscures the cutoff. The sparse upper tail causes a wide and thus seemingly-outlying last bin.

We quantify contribution as the number of tweets sent[7], so the lifetime contribution is the tweet count when the user becomes inactive. Following Wilkinson [134], a user that has not tweeted for six months (as of June 2013 when our lifetime contributions dataset was collected) is *inactive*.

Figure 6.2 plots the logarithmically-binned [147] empirical distribution. It is heavy-tailed, but decays more quickly in the upper tail than a true power law. The higher density in the last bin (~200 000 tweets) is due to Twitter's rate limits of 1000 tweets per day and 100 tweets per hour, because users that would occupy the upper tail (>200 000 tweets) are forced into this bin.[8] YouTube exhibits the same non-power law, upper tail cutoff [146], consistent with a common dynamic underlying both systems.

[7]One could instead consider retweets, replies, or direct messages, but obtaining data for these is more difficult.

[8]The rate limit means that the lifetime contribution distribution can be viewed as a censored [148] version of the "natural" distribution.

Table 6.2: Power-Law Exponents for Lifetime Contributions in Various Online Communities, Computed Incorrectly Using Equal-Count Binning

Contribution Type	α	**p-value**	**min. k**
Essembly votes†	1.47	0.59	3
Digg votes†	1.53	0.64	15
Twitter tweets	*1.54*	*0.96*	*12*
Bugzilla comments†	1.98	0.74	5
Essembly submissions†	2.02	0.25	7
Wikipedia edits†	2.28	0.69	10
Digg submissions†	2.40	0.04	15
Youtube submissions‡	2.46	—	—

† from Wilkinson [134]
‡ from Huberman, Romero, and Wu [146]

6.3.1 Critique of Previously-Reported Power Law Behavior

Surprisingly, the cutoff does not match the strong power law evidence reported for Bugzilla, Essembly, Wikipedia, and Digg [134]. We believe those systems do contain a similar cutoff, but it was obscured by the analysis methods used. We observe three weaknesses of the prior approach. First, the equal-count binning[9] method used obscures the upper tail behavior; logarithmic binning is preferred [147]. Second, maximum likelihood estimation, not binned regression, should be used for fitting [149]. Finally, the goodness-of-fit should be computed against the empirical distribution function (Kolmogorov–Smirnov or Anderson–Darling test) [149], not against binned data (the G-test).

The original datasets are unavailable[10], so we tested our hypothesis by applying the same methods to our Twitter data. As expected, equal-count binning, shown in the inset of Figure 6.2, hides the known cutoff. The G-test for a power law fit by regression to the improperly binned data indicates a good match (Table 6.2), despite the obvious mismatch in the real data. Clearly, these methods can obscure any underlying cutoff. Our results

[9] In equal-count binning, each bin is sized to contain the same number of samples and thus the same area under the density function. For B bins, the height of a bin b_i is computed as $B/w(b_i)$, where $w(b_i)$ is the width of b_i.

[10] Emails to the author bounced as undeliverable.

Table 6.3: Parameters for Distributions of Lifetime Tweets

Distribution		**Parameters**	
Name	PMF	(fit by MLE)	
Power Law	$\frac{1}{\zeta(\alpha, x_{\min})} \cdot \frac{1}{x^{\alpha}}$	α	1.54
		$x_{\min}$	12.00
Type-II Discrete Weibull [150]	$\frac{c}{x^{1-\beta}} \prod_{n=1}^{x-1} \left(1 - \frac{c}{n^{1-\beta}}\right)$	β	0.17
		c	0.32

are consistent with Bugzilla, Essembly, Wikipedia, and Digg contributions containing the same cutoffs as Twitter and YouTube, but the original data would be needed to prove this conclusion.

6.3.2 Lifetime Tweets Follow a Weibull Distribution

If the distribution is not power law, what is it? Examining the *hazard function*, or probability that a user who has made x contributions quits without another, provides the answer. Shown in Figure 6.3, the hazard function is an obvious power law. Wilkinson referred to this behavior in other online communities as *participation momentum* [134]; we will return to that interpretation later.

The power law hazard function $\frac{\alpha-1}{x^{1-\beta}}$ is that of the Weibull distribution[11], for continuous support. For discrete support, the distribution with a power law hazard function is called a Type II Discrete Weibull[12] [150] and has mass function

$$\Pr(X = x) = \frac{\alpha - 1}{x^{1-\beta}} \prod_{n=1}^{x-1} \left(1 - \frac{\alpha - 1}{n^{1-\beta}}\right). \tag{6.6}$$

A maximum likelihood fit to the lifetime contribution data yields $\beta = 0.17$ and $\alpha = 1.32$, as shown in Figure 6.2. The upper tail deviates slightly, which we attribute to Twitter's rate

[11]The Weibull distribution is sometimes called the *stretched exponential*.

[12]The much more common Type I Discrete Weibull [151] instead preserves the exponential form of the complementary cumulative density function.

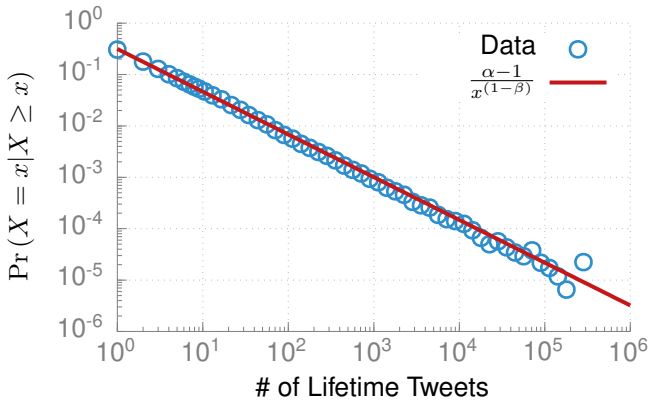


Figure 6.3: The probability that a user who has sent x tweets quits without sending another, i.e., the hazard rate. The decreasing trend suggests a sort of momentum; the more times a user has tweeted, the more likely he is to tweet again. The power law parameters are calculated from Table 6.3, not fit to the data.

limit policy. Some users that would have tweeted more than ~200 000 times were artificially limited to fewer tweets, increasing the weight in that portion of the upper tail.

6.3.3 Interpreting the Hazard Function as Participation Momentum

Wilkinson [134] used a notion of participation momentum to explain the power law hazard function. For his assumed power law distribution, $C\frac{1}{x^{\alpha}}$, the hazard function is $\frac{\alpha-1}{x}$ and α can be seen as a metric for the effort needed to contribute. Higher required effort leads to a higher probability of quitting. Table 6.2 shows the α's for several systems. Intuitively, tweeting seems more taxing than voting on Digg stories but less so than commenting on Bugzilla reports. And indeed, we find that $\alpha_{\text{Digg}} < \alpha_{\text{Twitter}} < \alpha_{\text{Bugzilla}}$.

Alternatively, the hazard function might be more directly related to account age than total contributions. To reject this possibility, we compared the Kendall tau rank correlations [152] between lifetime contributions, age, and average tweet rate (lifetime contributions/age). Unsurprisingly, age (i.e., longer life) correlates with increased lifetime contributions ($\tau = 0.4708$, $p = 0.00$, 95% CI [0.4690, 0.4726]). In contrast, the tweet rate is essentially uncorrelated with lifetime contributions ($\tau = -0.0067$, $p = 0.00$, 95% CI [-0.0085,

-0.0049]), indicating that the momentum function is not driven by age. If it were, the correlation would be strongly positive because faster tweeters would generate more tweets in their (independently determined) lifetimes. The strong negative relationship between tweet rate and age ($\tau = -0.5687$, $p = 0.00$, 95% CI [-0.5705, -0.5669]) further supports this conclusion. The hazard rate is determined by the current total contributions, so users with higher tweet rates must have shorter lifetime ages.

The hazard function we observe ($\frac{\alpha-1}{x^{1-\beta}}$ instead of Wilkinson's $\frac{\alpha-1}{x}$) invites additional thought. The new parameter β ($\beta = 0$ in Wilkinson's model) models momentum gain—a higher β translates to more momentum gain per contribution. For example, one could imagine that β reflects the effect of feedback. Positive (negative) viewer-generated feedback like retweets and replies in Twitter or comments and view counts in YouTube might accelerate (decelerate) momentum gains relative to systems without such visible feedback, like Digg votes or Wikipedia edits.[13] Refinement of this interpretation is a promising area for future work.

In summary, lifetime contributions in Twitter are driven by a power law hazard function $\left(\frac{\alpha-1}{x^{1-\beta}}\right)$ viewed as participation momentum. α reflects the effort needed to contribute and β the amount of feedback provided by system. The power law momentum leads to a Type II Discrete Weibull distribute for lifetime contributions. This dynamic holds across a variety of online communities [134, 146].

6.4 Distribution of Tweet Rates

The distribution of tweet rates is arguably the most important statistic for microblogging system design. An architecture designed for uniform messaging rates across the network will struggle with a heavy-tailed rate distribution. In this section, we describe an analytical model

[13] Wilkinson's reported results are consistent with this hypothesis. The contribution types with the most visible feedback—Essembly and Digg submissions—show little support for a power law, with p-values of 0.25 and 0.04. $\beta > 0$ would explain the non-power law behavior. The distribution for YouTube by Huberman et al. also shows a cutoff [146] consistent with a hazard function with $\beta > 0$.

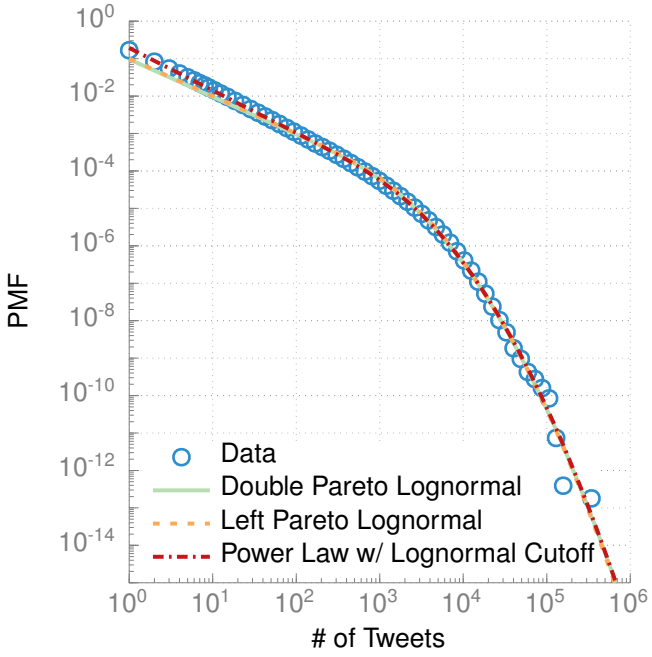


Figure 6.4: Distribution of tweets per user for the four month period from June through September 2012.

and generative mechanism for the rate distribution and reject a model previously proposed for telephone call rates. Although we are most interested in the tweet rate distribution, we model the easier-to-consider tweet count distribution. The former is easily recovered by dividing out the 4-month sampling duration.

6.4.1 An Analytical Approximation of the Tweet Rate Distribution

Figure 6.4 plots the logarithmically-binned empirical tweet distribution. It is heavy-tailed, consistent with other forms of authorship [153]. The tails form two different regimes meeting at $X = \sim 2000$, each heavy-tailed but with different exponents. We show in Subsection 6.4.3 that this phase change is a dynamic effect related to the sample period length (i.e., four months)—the crossing point increases with the square of the sample period length.

Simulating microblogging performance and comparing rates across communication systems benefits from a closed-form of the distribution. The forthcoming generative model in Subsection 6.4.3 is not analytically tractable, so we describe an analytical approximation

first. Figure 6.4 suggests a cutoff power law, but the upper tail is heavier than the common exponential cutoff [149]. Instead, the cutoff appears lognormal, suggesting the following density function[14],

$$p(x) = cx^{-\beta}\Phi^c\left(\frac{\ln x - \mu}{\sigma}\right), \tag{6.7}$$

where Φ^c is the complementary CDF of the standard normal distribution and c is a normalizing constant. The maximum likelihood fit is shown in Figure 6.4, with $\beta = 1.13$, $\mu = 7.6$, $\sigma = 1.06$, and $c = 0.19$. The lognormal cutoff shape is seen by noting that

$$\Phi^c(z) \propto \mathsf{erfc}\left(\frac{z}{\sqrt{2}}\right) \text{ and } \mathsf{erfc}(z) \approx \frac{1}{\sqrt{\pi}}\frac{e^{-z^2}}{z} \text{ for } z \gg 1,$$

leading to the approximately lognormal form

$$\Phi^c\left(\frac{\ln x - \mu}{\sigma}\right) \lessapprox \frac{\sigma}{\ln x - \mu} e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}} \text{ for } \frac{\ln x - \mu}{\sigma} \gg 1. \tag{6.8}$$

The power law exponent in the lower tail is β, the phase change to the cutoff regime occurs at $\exp(\mu)$, and the upper tail steepness is controlled by σ.

6.4.2 The Distribution is Not Double Pareto–Lognormal

At first glance, Figure 6.4 appears to be Double Pareto-Lognormal (DPLN), a recently-discovered distribution that has found wide-spread popularity across many fields, perhaps due to its clear generative interpretation [154]. Seshadri et al. suggested its use for communication rates, specifically call rates in a cellular network, interpreting the generative process as evolving *social wealth* [135]. However, in this section we show that the DPLN does not correctly capture the lower tail behavior of tweet rates (or call rates). In the next section, we describe a different mechanism to explain the shape.

We first summarize the origin of the DPLN distribution [154]. Given a stochastic process

[14] We use a continuous model for simplicity. The integral data can be viewed as a rounded version of the product of the true tweet rate and sampling period.

X evolving via Geometric Brownian motion (GBM)

$$\mathrm{d}X = \mu X + \sigma X \,\mathrm{d}W, \tag{6.9}$$

where W is the Wiener process, with lognormally distributed initial state, $\log X_0 \sim \mathcal{N}(\nu, \tau^2)$, then X_t is also lognormally distributed, $\log X_t \sim \mathcal{N}(\nu + \frac{\mu-\sigma^2}{2}t, \tau^2 + \sigma^2 t)$. If the observation (or killing) time $t \triangleq T$ is exponentially distributed, $T \sim \mathrm{Exp}(\lambda)$, then the observed (or final) state has DPLN distribution, $X_T \sim \mathrm{DPLN}(\alpha, \beta, \nu, \tau)$, where $\alpha > 0$ and $-\beta < 0$ are the roots of the characteristic equation

$$\frac{\sigma^2}{2}z^2 + \left(\mu - \frac{\sigma^2}{2}\right)z - \lambda = 0. \tag{6.10}$$

Seshadri et al. [135] proposed that the number of calls made by an individual reflects an underlying *social wealth* that evolves via such a GBM. For an exponentially growing population, the age distribution of the sampled users is exponential and the resulting distribution of calls (or social wealth) will be DPLN. This model seems qualitatively reasonable for Twitter as well, but cannot capture the correct power law exponent in the lower tail (see Figure 6.4). The call distribution data exhibits a similar mismatch, challenging the model's suitability there as well.

The density function of $\mathrm{DPLN}(\alpha, \beta, \nu, \tau)$ is

$$f(x) = \frac{\beta}{\alpha+\beta}f_1(x) + \frac{\alpha}{\alpha+\beta}f_2(x), \tag{6.11}$$

where

$$f_1(x) = \alpha x^{-\alpha-1}A(\alpha, \nu, \tau)\Phi\left(\frac{\ln x - \nu - \alpha\tau^2}{\tau}\right), \tag{6.12}$$

$$f_2(x) = \beta x^{\beta-1}A(-\beta, \nu, \tau)\Phi^c\left(\frac{\ln x - \nu + \beta\tau^2}{\tau}\right), \tag{6.13}$$

$$A(\theta, \nu, \tau) = \exp\left(\frac{\theta\nu + \theta^2\tau^2}{2}\right), \tag{6.14}$$

and Φ and Φ^c are the CDF and complementary CDF of the standard normal distribution. f_1 and f_2 are the limiting densities as $\alpha \to \infty$ and $\beta \to \infty$, respectively, and are called the *right Pareto lognormal* and *left Pareto lognormal* distributions.

Two observations stand out. First, the distribution is Pareto in both tails, with minimum slope of -1 in the lower. Second, the left Pareto lognormal form is nearly equivalent to our expression Equation 6.7, which differs only by accommodating lower tail exponents below -1. Figure 6.4 shows maximum likelihood fits of both the DPLN and left Pareto lognormal distributions. The lower tail is steeper than allowed by the DPLN ($-1.13 < -1$) and fits poorly. The call distribution data shows a similar mismatch. Although the DPLN is widely applicable, it does not model these communication patterns. Our model from the following section should better fit the call data [135] as well.

In the upper tail, both distributions fit equally well (i.e., a likelihood ratio test does not favor either fit). The data are insufficient to distinguish a lognormal from a power law upper tail, a common issue [149]. We favor the lognormal form for Equation 6.7 because it is simpler (i.e., has fewer parameters) and most real world "power laws" exhibit some cutoff [149].

6.4.3 An Urn Process Generating the Tweet Rate Distribution

In this section we develop an urn process to describe tweet distribution in Figure 6.4. The phase change is a dynamic effect governed by the sampling period. As the period increases, the distribution approaches that of the lower tail—approximately Pareto with exponent -1.13. In practical terms, high-rate tweeters are much rarer in a finite sample than the asymptotic distribution would predict.

Figure 6.5 shows the distribution for two sample periods, illustrating the dynamic phase change. The lower tail extends further with the longer period. Degree distributions in

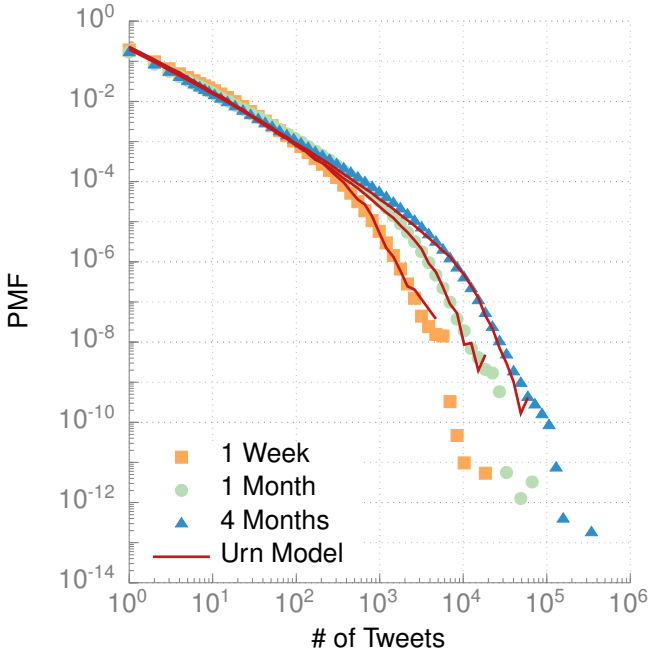


Figure 6.5: Distribution of tweet counts over various sample periods, showing the time-dependent cutoff. The asymptotic distribution is Pareto. Traces for the urn model describing this effect were obtained by simulation.

growing networks evolve similarly. Although simple preferential attachment of new nodes leads to a straight power law [155], when existing nodes also generate new edges via preferential attachment, the distribution is double Pareto (with exponents -2 and -3) with a time-dependent crossing point ($k_c = [b^2 t(2 + \alpha t)]^{1/2}$) [7].[15] A similar model describes the tweet distribution.

Consider the evolution of the sample of tweets. Users join the sample upon their first tweet (during the sample period) and then continue to produce additional tweets at some rate. Discretize time relative to new users joining the sample, i.e., one user joins at each time step so there are t users at time t. Let $k(s, t)$ be the (expected) tweet count at time t for the user first observed at time s. Assume new tweets are generated at a constant average rate c, i.e., ct new tweets appear at each time step, distributed among existing users with frequency proportional to $A + k(s, t)^\alpha$. A is the initial attractiveness and α is the non-linearity of the

[15] In a network that allows self-edges, the exponents are -3/2 and -3 with crossing time $k_c \approx \sqrt{ct}(2 + ct)^{3/2}$ [156].

preference [157]. The resulting continuum equation[16] is

$$\frac{\partial k(s,t)}{\partial t} = (1+ct)\frac{A + k(s,t)^\alpha}{\int_0^t A + k(u,t)^\alpha \, \mathrm{d}u} \tag{6.15}$$

An analytical solution exists when $A = 0$ and $\alpha = 1$ [8], but for the general case we resort to Monte Carlo simulations. Figure 6.5 shows the close match to the empirical density when $A = 1$ and $\alpha = 0.88$.[17] Assuming the power law form of the asymptotic density, $p(k) \propto k^{-\beta}$, the power law form of the rate distribution can be recovered. Taking λ as the tweet rate and noting that $\lambda \propto k^{-\alpha}$ when $k \gg A$, then

$$p(\lambda) = p(k^{-\alpha}) \propto \frac{1}{\alpha}\lambda^{-\frac{-1+\alpha+\beta}{\alpha}}. \tag{6.16}$$

Thus, for α close to 1, the power law exponent recovered from Figure 6.4 slightly overestimates that of the tweet rate.

Relating back to the analytical approximation of Equation 6.7, μ is related to ct by $\mu \approx 1.32 \ln(ct) + 0.56$. $\beta = 1.13$ and $\sigma = 1.06$ are constants best determined by fitting.

6.4.4 Distributions of Retweeter and Retweetee Rates

The retweet and retweetee rates show a similar dynamic behavior in Figure 6.6. The retweet behavior differs only in the average rate c, which is about $2\times$ lower. The retweetee distribution exhibits two interesting differences. First, it extends further to the right, indicating that retweets of popular users outnumber tweets from extensive users. Second, the slopes of the power law regimes are more consistent with a pure preferential attachment process (i.e., $\alpha = 1$). The retweetee rate comes directly from a preferential attachment process—initial retweets increase exposure, begetting additional retweets—and thus should match the linear form seen in other systems. The power law form of the tweet and retweet rates describes the

[16] We use the notation and continuous approximation of Dorogovtsev and Mendes [8].

[17] Parameters were chosen by a coarse, manual exploration of the space. Fine-tuning might further improve the fit.

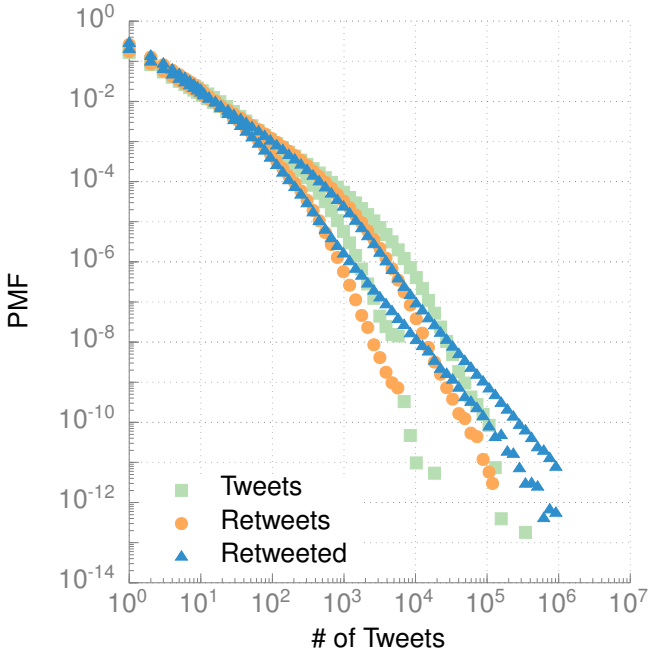


Figure 6.6: Distributions for tweets sent, retweets sent, and times retweeted for the 1 week and 4 month samples. All categories show similar time-dependent phase changes, suggesting the same underlying mechanism. Retweets differ from tweets only in a lower average rate (parameter c in the urn model).

underlying propensity to tweet, but without the same generative interpretation.

6.5 Distribution of Intertweet Durations

Arrival processes in communication systems are traditionally assumed to be Poisson [158], but *per-individual* interval distributions for various activities including email, printing, and telephone calls are heavy-tailed [136, 159, 160]. We show that this same behavior holds in Twitter, with our analysis mirroring that of Candia et al. for telephone calls [136] to enable easy comparison. The SNAP tweet dataset is used for this analysis.

We group the users by their total tweets to isolate the effects of differing tweet rates. Figure 6.7 plots the empirical distributions. Scaling by the group's average interevent time (Δt_a) collapses the distributions to a single curve, shown in Figure 6.8. This universal trait is also found in email and telephone systems [136, 161]: the distribution is described by

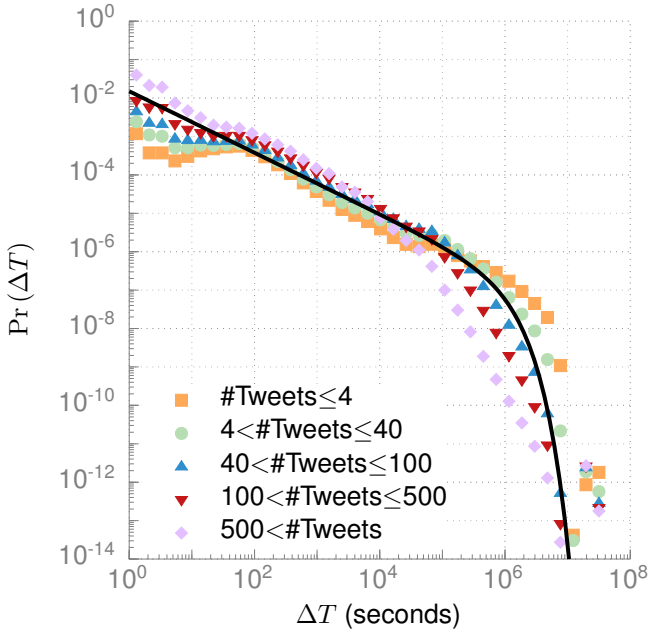


Figure 6.7: The interevent distributions with users grouped by number of tweets for the three month period covering June through August 2009. The line is a best-fit power law with exponential cutoff.

$\Pr(\Delta T) = \frac{1}{\Delta T_a} F\left(\frac{\Delta T}{\Delta T_a}\right)$, where $F(\cdot)$ is independent of the average rate. The best-fit cutoff power law is

$$\Pr(\Delta T) \propto (\Delta T)^{-\alpha} \exp\left(-\frac{\Delta T}{\tau_c}\right), \tag{6.17}$$

with exponent $\alpha \approx 0.8$ and cutoff $\tau_c \approx 8.1\,\mathrm{d}$, shown as the black line in Figure 6.8. ΔT_a is taken as the whole population average here.

6.6 Characteristics of the Retweet Graph

The natural and explicit network in Twitter—the social graph in which a directed edge represents the *follower* relationship—has been well-studied. Kwak et al. first reported on basic network properties like degree distribution, reciprocity, and average path length [122], and later works have studied these and other characteristics in more detail [123–126]. However, an alternative, implicit network—the retweet graph in which a directed edge

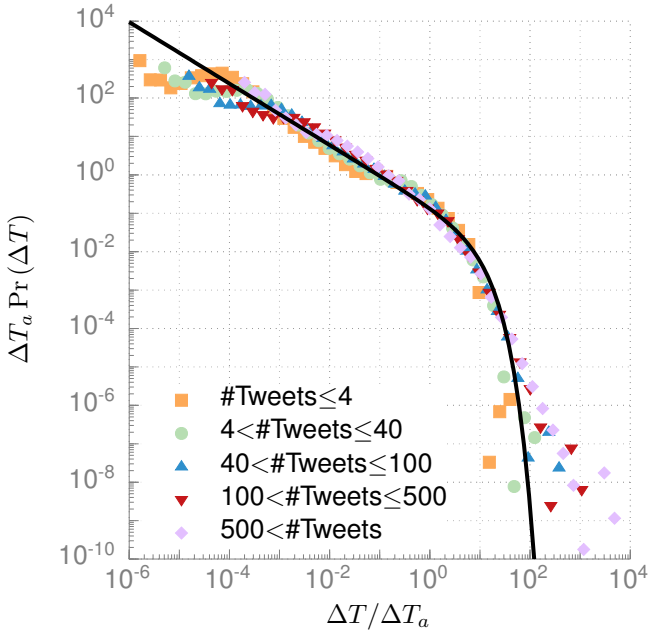


Figure 6.8: The interevent distributions of Figure 6.7 collapse when scaled by the group's average interevent duration, ΔT_a. The line is a best-fit power law with exponential cutoff.

indicates that the source retweeted the destination—has been neglected. We conduct the first characterization of the retweet graph and confirm that it, like many real-world networks, is small-world and scale-free. The reported metrics are useful for generating random retweet graphs using general parametric models like R-MAT [162] ($a = 0.52$, $b = 0.18$, $c = 0.17$, $d = 0.13$) or other specific generative models [163].

We pay particular attention to contrasting the social following[18] and retweet graphs. Intuitively, they should be similar because retweets are usually sent by followers. However, we conjecture that the retweet graph more closely models the real-world social and trust relationships among users, because it derives from a more forceful action—not just listening to others' ideas, but forwarding them to one's own friends. Using the follower graph as a trust proxy has been proposed for applications ranging from spam filtering [164–166] to Sybil detection [106, 167]. We conjecture that the retweet graph is a better choice and provide some supporting evidence. Full treatment of this conjecture is beyond our scope.

[18]The social following graph is simply the social follower graph with the edge direction reversed to match that of the retweet graph.

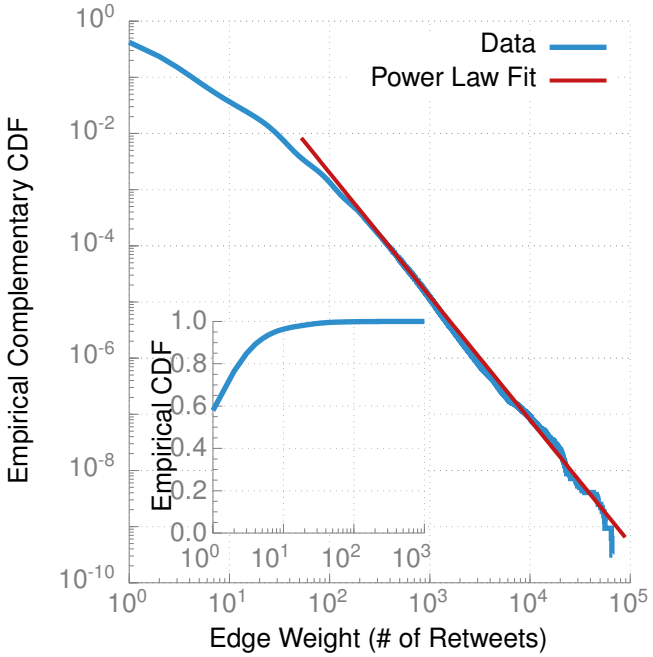


Figure 6.9: Distribution of number of edge weights in the retweet graph, corrected using the EM method. A directed edge indicates that one user retweeted another and the weight is the number of such retweets.

6.6.1 Analyzing a Random Subsample of the Retweet Graph

The retweet graph is constructed from our largest dataset, the 10% sample, and thus does not contain all edges. An edge is included with probability proportional to the number of retweets sent along it. However, 60% of edges have a single retweet and 98% have fewer than 10 (see Figure 6.9), so for simplicity we assume each edge is included with 10% probability. Many measured properties in an edge-sampled graph differ from the original graph. When possible, we use the EM-based method from Subsection 6.2.5 to correct our results. When not, we estimate the errors using the literature on sampled graphs [139, 141, 168].

6.6.2 Degree Distributions

We begin with the in- and out-degree distributions. The in-degree k^i_{in} of a node i is the number of unique users who retweeted i and the out-degree k^i_{out} is the number of unique users retweeted by i. The average in-degree $\langle k_{\text{in}} \rangle \triangleq N^{-1} \sum_{i \in V} k^i_{\text{in}} = 88.4$ and the similarly-

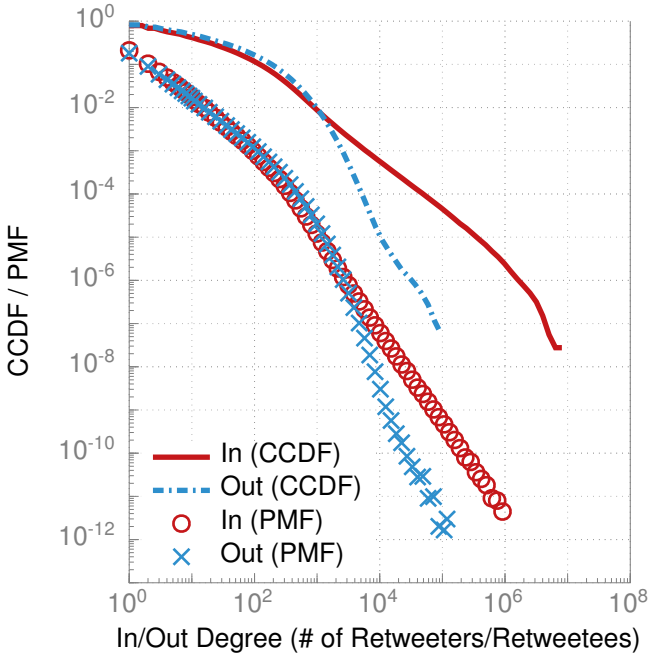


Figure 6.10: In and out degree distributions for the retweet graph. Both exhibit the double-Pareto behavior common to evolving networks [7, 8]. In the upper tail, the in-degree power-law exponent is 2.2 and 3.75 for the out-degree.

defined average out-degree $\langle k_{\text{out}} \rangle = 74.3$. V is the set of nodes and N their cardinality. In reality $\langle k_{\text{in}} \rangle = \langle k_{\text{out}} \rangle$; the observed difference is an artifact of the EM-based population estimation. The degree standard deviations are $\sigma_{\text{in}} = 4187.3$ and $\sigma_{\text{out}} = 228.4$. Higher in-degree variance is expected because, as with real-world networks [141], *popularity* (the number of users who retweeted an individual) is more variable than *extensivity* (the number of users an individual retweeted).

The distributions, shown in Figure 6.10, are similar to the social following graph [122]. Both are heavy-tailed and exhibit the same two-phase power law common to such networks. Similarly to the tweet rate distribution (Subsection 6.4.3), the two phases are a dynamic effect arising from two forms of evolution in the graph [7, 8]—the addition of new nodes and preferentially-attached new edges between existing nodes. The outgoing (incoming) node i for a new edge is selected with relative probability $d_{\text{out}}(i) + \delta_{\text{out}}$ ($d_{\text{in}}(i) + \delta_{\text{in}}$), where δ_{out} and δ_{in} are the initial attractiveness constants and $d(\cdot)$ returns the node degree. Bollobás

et al. elucidate this process for a general context [163].

The power law exponents are determined by δ_{out} (δ_{in}). The lower tails are similar with $\alpha \approx 1.3$. In the upper tail, $\alpha_{\text{out}} = 3.75$ and $\alpha_{\text{in}} = 2.2$. a_{in} matches the followers graph (2.3) [122] and is in the range of most real-world networks (2–3). α_{out} exceeds that range because extensivity is not inherently preferential (like popularity).

6.6.3 Reciprocity

Reciprocity is the fraction of links that are bidirectional. Many social networks have high reciprocity—most relationships are bidirectional (68% in Flickr [169] and 84% on Yahoo! 360 [170]). In the Twitter follow graph, reciprocity is lower at just 22.1% [122]. If retweeting is more discriminating than following, the retweet reciprocity should be lower. Indeed, it is just 11.1%.[19] Higher reciprocity in the follower graph may stem from the popularity of follow-back schemes in which a user, in an attempt to gain followers, promises to follow back anyone who follows him. The low reciprocity suggests that using the retweet graph as a proxy for trust is promising. Although a malicious node can establish many outgoing links, it has little control over the incoming structure.

6.6.4 Average Shortest Path Length (Degree of Separation)

The real-world human social network has a small average shortest path length (APL) of about six, shown most famously by Stanley Milgram [171, 172]. Many online networks are similar [173, 174], but the social followers graph is denser with an APL of 4.12 [122]. Kwak et al. attribute this difference to Twitter's additional role as an information source. Edges are more dense because users follow *both* social acquaintances and sources of interesting content.

[19] We estimated the distribution of all non-zero pairwise edge weight tuples (the number of retweets in both directions) from the 10% sample using the EM algorithm. The fraction that are non-zero in both directions is the reciprocity.

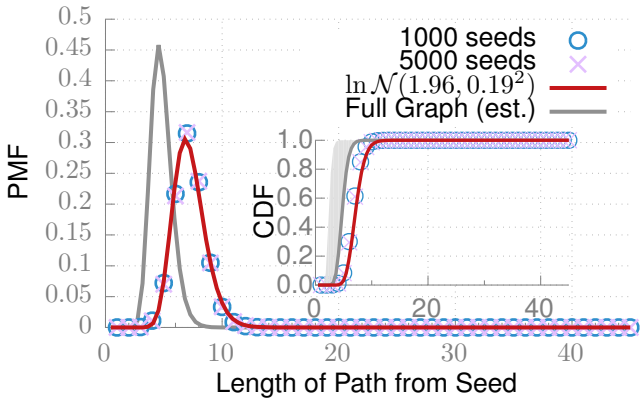


Figure 6.11: Distribution of average path length (degree of separation) in edge-sampled retweet graph. The gray line is the estimated distribution for the full graph.

We determined the path length distribution of the 10% edge-sampled graph by computing all shortest paths for both 1000 and 5000 random starting nodes. The obtained distributions (shown in Figure 6.11) overlap, indicating a sufficient sample size. Lee et al. showed that edge sampling increases the APL by 1.5–$3\times$ (the gray range in the inset plot) depending on the graph structure [139]. We use $1.5\times$, determined by sampling the followers graph[20], to estimate the full distribution (grey line in plot). The estimated APL is 4.8 and the 90th-percentile or *effective diameter* [175] is 8.5. The difference from the followers graph is within estimation error.

The best-fit distribution (solid line in plot) is log-normal[21] with $\mu = 1.5$ and $\sigma = 0.27$. This differs from undirected Erdös-Rényi (ER) graphs, for which the limiting distribution is Weibull [176], but we do not know of similar theoretical results for directed graphs.

6.6.5 Assortativity (Node Degree Correlation)

Degree assortativity—the tendency of nodes to connect with others of similar degree—summarizes the structural characteristics that in part determine how content (e.g., retweets or disease) spreads and resilience to node removal [177]. In an assortative network, content

[20]The 2009 crawl [122] is complete, so we compared the true statistic against that of a 10% subsample.

[21]We compared with the Weibull, Gumbel, Fréchet, and encompassing generalized extreme value distributions.

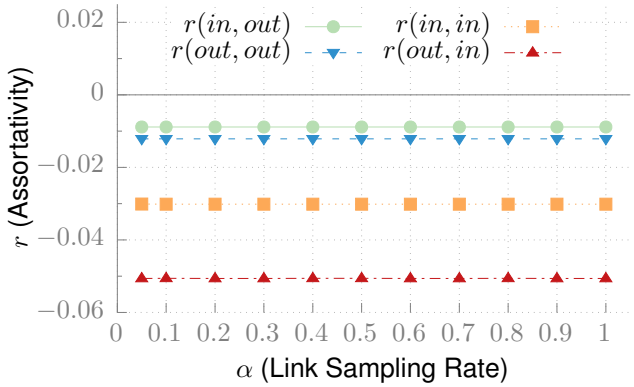


Figure 6.12: Directed assortativities r as a function of edge sampling rate. Edge sampling does not affect assortativity because all node degrees are sampled independently and identically.

easily propagates through a connected component of tightly clustered, high degree nodes that is resistant to node removal, but may not reach the low degree boundary of the network. Conversely, a disassortative network has a larger connected component so content propagates further, but can be partitioned by the removal of a high degree node.

For undirected networks, assortativity is simply the Pearson correlation between the degrees of adjacent nodes. The concept generalizes to directed networks by considering all possible directional degree pairs as separate assortativity metrics [178], $r(in, in)$, $r(in, out)$, $r(out, in)$, $r(out, out)$, again using the Pearson correlation

$$r(\alpha, \beta) \triangleq \frac{\langle k_\alpha^i k_\beta^j \rangle - \langle k_\alpha^i \rangle \langle k_\beta^j \rangle}{\sigma_{k_\alpha} \sigma_{k_\beta}} \tag{6.18}$$

where $\alpha, \beta \in \{in, out\}$, k_α^i (k_β^j) is the α-degree (β-degree) of source (destination) node i (j), the averages $\langle \cdot \rangle$ are taken over all directional edges $(i \to j)$, and σ_{k_α} (σ_{k_β}) is the variance of the α-degree (β-degree).

We characterize and contrast these metrics for both the Twitter social following graph [122] and retweet graph. Edge sampling impacts the degrees of all nodes identically and thus does not effect assortativity (see Figure 6.12) [139].

Figure 6.13 plots the assortativities for both networks. Although most real-world social

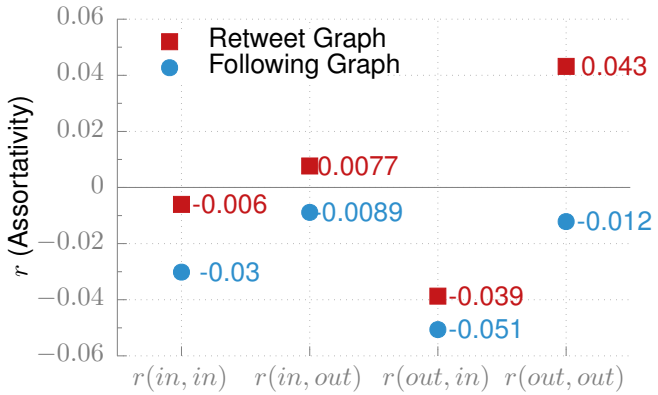


Figure 6.13: Directed assortativity r of the retweet graph and the social following graph. The retweet graph has higher assortative, more consistent with real world social networks than most online social networks.

networks are assortative [177], online social networks are instead disassortative [179]. The social followers graph is no exception, showing weak disassortativity across all measures. In contrast, the retweet graph is more assortative across all measures. It is near-neutral for both $r(in, \cdot)$ metrics, indicating independence between one's own retweet behavior and the number of retweets received. This is consistent with the graph containing useful trust information, because a user cannot influence the quantity of retweets received by selectively retweeting popular ($r(in, in)$) or extensive ($r(in, out)$) users. The high (out, out) assortativity is more consistent with real-world social networks and indicates that extensive retweeters retweet each other. Interestingly, they are not tightly clustered (or the (in, out)-assortativity would be higher).

In Twitter, tweets propagate to followers, so the social graph disassortativity is helpful. The connected component is larger and tweets disseminate further more quickly. Increased susceptibility to node failure is acceptable in a centralized system. In a decentralized system that relies more heavily on the retweet graph for propagation, e.g., Shout [132], the resilience to node failure implied by its neutral and positive assortativities would instead be helpful.

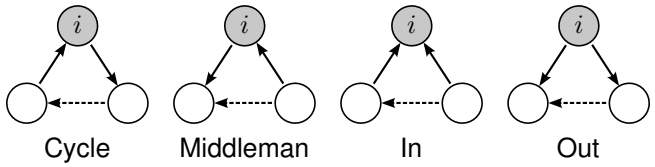


Figure 6.14: The four types of open (solid edges) and closed (solid and dashed edges) directed triplets used for cluster analysis. A vertex can form up to eight such triplets with each pair of neighbors, two of each type. The clustering coefficient $C_{\beta\in\{\text{cycle, middleman, in, out}\}}$ is the fraction of β-triplets (open and closed) that are closed.

6.6.6 Clustering Coefficient

A clustering coefficient quantifies the tendency of neighboring nodes to form highly connected clusters. Many real-world networks exhibit tighter clustering than would be expected in similar random graphs [173]. We consider the *global clustering coefficient*[22], defined for undirected graphs as

$$C \triangleq \frac{3N_{\triangle}}{N_3}, \tag{6.19}$$

where N_3 is the number of open *or* closed triplets (three vertices connected by two or three edges) and $N_{\triangle}$ is the number of closed triplets (3-vertex cliques). Unlike the alternative *local clustering coefficient*, this definition is suitable for networks with isolated nodes [180]. In essence, C gives the probability that any two neighbors of a node are themselves connected.

Following the approach introduced by Fagiolo for the local clustering coefficient [181], we extend the metric to directed graphs by separately considering the four types of directed triplets, shown in Figure 6.14. The four clustering coefficients $C_{\beta\in\{\text{cycle, middleman, in, out}\}}$ are the fraction of β-triplets that are closed.

An estimator from the sample clustering coefficient of an α-edge sampled graph ($\alpha = 0.1$ for us) is

$$\widehat{C} \triangleq \frac{1}{\alpha}\overline{C}, \tag{6.20}$$

seen by noting that a triplet is included in the sample with α^2 probability and as a closed

[22] Sometimes called the *transitivity* or *transitivity ratio*.

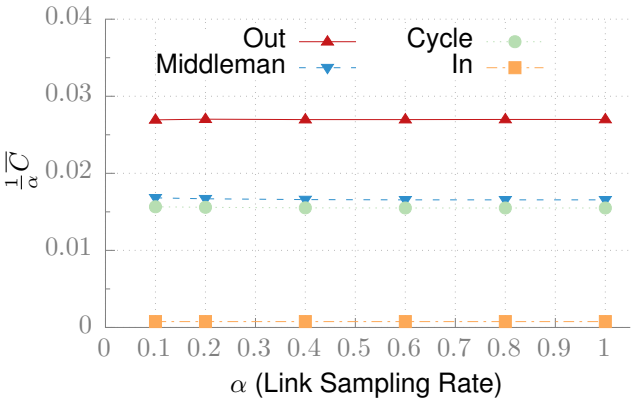


Figure 6.15: The clustering coefficient estimator $\widehat{C} \triangleq \frac{1}{\alpha}\overline{C}$ as a function of edge sampling rate on the social "following" graph. Although potentially biased, the estimator is quite accurate for such graphs.

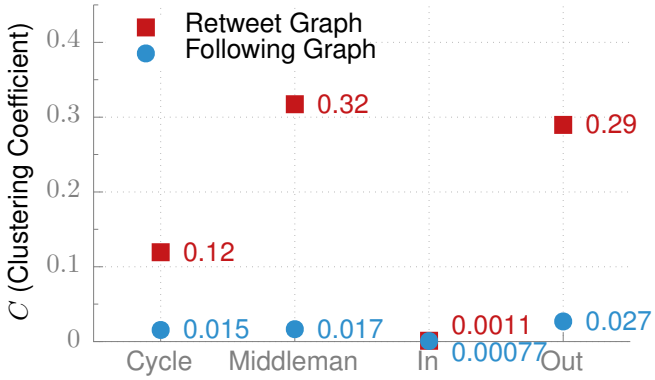


Figure 6.16: Clustering coefficients for the social "following" graph and the retweet graph. Clustering is significantly more prominent in the retweet graph and more consistent with real-world social networks.

triplet with α^3 probability. This estimate is biased, because the triplets are not independent and edges can be concentrated towards open (or closed) triplets. In practice however, it performs well on large samples, as shown in Figure 6.15 for the social following graph.

Figure 6.16 plots the results for both the social and retweet graphs. The former has low clustering, but clustering in the retweet graph is significant in metrics except *in*. *Cycle* is the only fully (transitively) connected triplet type, and thus *cycle*-clustering should best reflect true clustering in the underlying social groups and interest topics. The higher clustering in the retweet graph indicates that retweet relationships are more concentrated than *following*

relationships, consistent with our hypothesis of higher trust.

Although the *middleman*, *in*, and *out* cycle are all rotations of the same basic non-transitive triplet, their coefficients differ due to the non-uniform degree distribution. C_{in} is low because the majority of (in, in) edge pairs are from a few popular users who are retweeted by many otherwise-unrelated users. The high $C_{\text{middleman}}$ and C_{out} coefficients are reflections of the same phenomenon—transitive retweeting. User f retweeting user i's retweet of user a is recorded by Twitter as f retweeting a (hence the name middleman). Often f will also retweet some of i's original content, closing the triplet. In the out case, node i plays the role of f instead of the middleman. Surprisingly, such transitive retweeting happens frequently ($C_{\text{middleman}} = 0.32$ and $C_{\text{out}} = 0.29$). In other words, 30% of these possible two-degree retweet relationships exist.

6.6.7 Summary

We have confirmed that the retweet graph is scale-free and small-world, like many social networks. Interestingly, its clustering and assortativity are closer to real-world networks than typical online networks, indicating that it may better capture real-world relationships and have application as a proxy for trust. Full treatment of this conjecture is beyond our scope. The scale-free, small-world confirmation enables the generation of random instances, e.g., using R-MAT [162], for empirical study. We use this approach in Section 6.8 to evaluate the use of connectivity in the retweet graph to detect spammers.

6.7 Implications for the Design of Decentralized Microblogging Architectures

The preceding sections characterized tweet behavior—total quantity, average rate, and interevent time—and the retweet graph structure. Although interesting in their own right, in this section we discuss a particular application—the implications for the design of

performance-constrained, decentralized microblogging platforms like Shout [132]. In such systems, bandwidth and energy—scarce resources—must be carefully allocated to achieve some notion of fairness. We discuss implications for such allocation strategies.

Power Law Participation Momentum: Most users quit after a few contributions, so greedy allocation of resources to new users is wasteful. For example, a routing scheme prioritizing messages from users with more contributions would implicitly direct bandwidth away from temporary users.[23] The known power law form of the momentum function enables the design of optimal allocation strategies. For example, consider storing old messages by distributing them across participating nodes. Nodes with more contributions are more reliable (less likely to leave the network) and thus require a lower storage replication factor. These failure probabilities can be easily modeled.

Heavy-Tailed Rate Distribution: The two-phase tweet rate distribution has implications for short-term message delivery and long-term message storage. The message generation rate may be modeled as lognormal—messages are naturally better-distributed in the network than a power law would suggest, reducing points of congestion and better balancing bandwidth use. In the long term, however, the average tweet rates follow the asymptotic power law with its much heavier tail, posing issues for archiving and retrieval of tweets. For example, sharding messages across nodes by author will result in a few nodes storing and serving the majority of the archived content. The archiving system must be designed to handle the power law distribution.

Heavy-Tailed Interevent Distribution: Simulations and other performance analysis must use heavy-tail distributions for the interevent times. Standard Poisson distributions will grossly underpredict these times, increasing simulated congestion and resulting in over-provisioned designs.

Small-World, Assortative, Clustered Retweet Graph: In a centralized platform, a single entity can moderate bad behavior, reject spammers, and ensure fair division of resources.

[23] We do not consider how malicious users might manipulate such schemes, but resistance to such attacks would be important for any practical protocol.

Participants in a decentralized platform must perform these same tasks themselves without implicit trust in others. The implicit retweet graph seems to encode some information about the real-world relationships of users that could be inferred for such purposes. The higher assortativity is more indicative of a real world network than a social network and the high clustering implies that users have some commonalities around which they gather. We explore this direction in the next section, using spammer detection via connectivity in the retweet graph as an example.

6.8 Leveraging the Retweet Graph for Spammer Detection

Spam is a problem for many communication platforms [166, 182], but seems particularly concerning for a system like Shout. A malicious user can easily flood its one-hop neighborhood with a multitude of useless or spam shouts. Twitter, a centralized service, can decree what constitutes spam, use its full knowledge of user behavior to detect violators [164, 165, 183–187], and limit the creation of new accounts. In a decentralized system, however, the lack of a trusted authorities implies that participants must do their own spam filtering, either individually or collaboratively.

In this section, we use the analysis of a spam detection method as an example use case for the preceding characterizations of the retweet graph. Specifically, we employ synthetic graphs that mimic the structure of the retweet graph to characterize the performance of a retweet (or reshout) graph-based spam technique.

6.8.1 Possible Approaches to Spam Detection

Although at first glance Shout appears to be highly susceptible to spammers, the underlying distributed architecture provides some implicit protection. Shouts are carried beyond a one-hop radius only if manually reshouted. Conforming users will not reshout spam,

implying that the spammers must do so themselves. As such, the cost of spreading spam is proportional to the number of users impacted, either in time (if the spammer moves about the network) or equipment costs (if the spammer uses multiple transmitters at different locations). Even occasional waves of spam are still undesirable, so a method for filtering spam from one-hop neighbors as well is needed.

Detection approaches can focus either on individual messages, *spam detection*, or on the sender, *spammer detection*. The latter is most applicable to microblogs, because the short message lengths—less than 250 characters—make content analysis difficult [164]. Spammer detection takes two forms differentiated by the default presumption. *Blacklisting* assumes that users are not spammers until proven otherwise, while *whitelisting* presumes the opposite. The former is a non-starter in Shout because blacklisted accounts are easily and cheaply replaced. Some form of whitelisting is required.

The most obvious form of whitelisting—and one applicable to Shout—is explicit labeling of accounts. The initial assumption of guilt implies that the first messages from a new user will not be seen—other nodes will filter it as spam. To bootstrap around this problem, users may explicitly whitelist accounts they wish to hear. For example, new users should ask to be added to their friends whitelists, allowing their messages to be seen.

Explicit whitelisting is sufficient for Shout to be usable. Original content is only seen by one's explicitly whitelisted friends, but can still spread via reshout. Consider a user A who has whitelisted user B who has whitelisted C. A messages from C will initially only be seen by B (not A). If interesting enough though, B will reshout, making it visible to A as well. This approach essentially mimics the flow of information in Twitter, i.e., along edges in the social graph, but requires geographic proximity as well. Some initially whitelisted nodes might later begin spamming (either as an intentional bait-and-switch or due to device hijacking), but are easily manually blacklisted. The whitelist is built manually and thus is relatively short, so corrective blacklisting will not be overwhelming.

Explicit whitelisting will often be overly strict, however. Interesting or useful messages

from geographically-proximate users outside of one's social circle will, at best, appear only after being retweeted by a friend, possibly after significant delay. We would like to instead be able to automatically infer the whitelist label for such users. We require this inference to be decentralized, so the possible approaches fall in two categories. The first uses only locally available information, e.g., the local database of overheard shouts, to perform classification. The second can use some form of transitive trust, e.g., the whitelists from other nodes or implicit signals of trust like reshouts.

Many researchers have considered spam detection using (locally available, in Shout) attributes of tweets [164, 165, 183–187]. In particular, Benevenuto studied the classification performance of 60 tweet and tweeter attributes, ranging from hashtags per tweet to the ratio of followers to friends. Aside from the obvious inclusion of URLs and account age[24], the most sensitive attributes were related to social behavior—ratio of followers to friends, number of replies to messages, etc. Noting that spammers can easily alter the content of tweets, they suggest focusing on these harder-to-manipulate attributes for detection. Their proposed classifier has a 70% true positive rate (TPR) and a 4% false positive rate (FPR).

Other researchers have considered incorporating the classifications from other participants, a form of transitive trust. However, such protocols can be subject to Sybil attacks, in which an attacker creates many identities reporting falsified observations to out-vote the honest identities. As a defense, Yu et al. developed SybilGuard, a Sybil detection scheme for social networks based on the observation that most Sybil identities will be weakly connected to the network [105]. Most people do not "friend" fake accounts on social networks, so fake identities receive only a limited number of connections. We consider similar graph-based approaches for spam detection.

Song, Lee, and Kim [165] applied this observation to spam detection in Twitter using the followers graph. In particular, they consider two metrics in the graph: *distance*—measured as the shortest path between two nodes—and *connectivity*—measured via max-flow and

[24]Twitter actively removes spammer accounts, biasing the collected data.

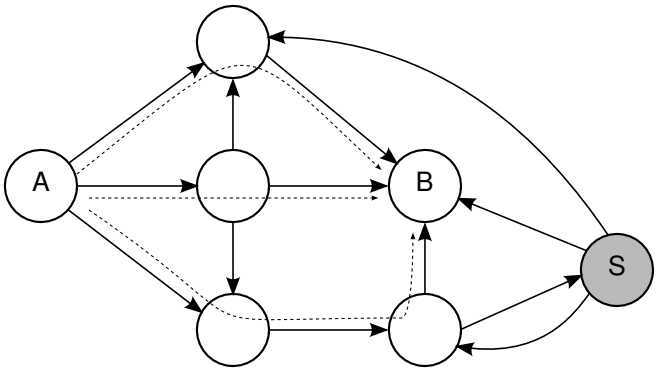


Figure 6.17: Portion of a retweet graph showing how spammers are less connected. Non-spammer B is connected to non-spammer A by three independent paths, the shortest of which has length two. Spammer S is connected by only a single length-three path.

random walk. A classifier over these attributes had a 95% TPR and a 4% FPR on their dataset. Including attributes like URLs per tweet improved the performance to a 99% TPR and 1% FPR.

Shout does not include explicit social relationships, so the followers graph cannot be used. Instead, we consider the implicit retweet graph. Intuitively, content from spammers will not be heavily retweeted, and thus they will be less connected to non-spammers in the graph, as illustrated in Figure 6.17. Node A is connected to non-spammer B by three edge independent paths, the shortest of which has length two. Spammer S, on the other hand, is only connected via a single path of length three. A separates spammers by classifying nodes based on their distance from and edge-independent connectivities (i.e., max-flow with unit-weighted edges) to itself.

Unfortunately, these techniques are not manipulation-resistant. Although they perform well on existing datasets, attackers that are aware of the classification method can adjust their own strategies to defeat detection. In particular, an attacker can "bait-and-switch"—initially broadcast non-spam content to gain reshouts (or followers) and then, once connected, switch to spreading spam. Although gaining an initial reshout (or follower) takes some effort, that one attacker account can then easily introduce more bait-and-switchers by reshouting their initial good content. Others who see that content will reshout, forming many edges back to

the new attacker nodes.

More abstractly, these techniques attempt to infer trust relationships from the behavior of the participants (e.g., reshouts) and then use transitive application of those trust relationships to classify spammers. Formal treatments of this problem domain (motivated by the design of recommender systems) [188–190] have revealed theoretical limitations on the power of such techniques. First, a tradeoff exists between detecting falsified reports and believing honest reports—any technique that limits the influence of falsified reports will also ignore some of the honest reports [189]. Second, safely employing transitive trust relationships implies some identities that would have been accepted as non-spammers (i.e., had a high enough trust balance) in by a direct trust protocol will be rejected as spammers (i.e., will not have a high enough trust balance) [190]. In Subsection 6.8.5, we summarize how our spam detection problem can instead be treated in manipulation-resistant fashion, although still subject to those two impossibility results.

6.8.2 Spam Detection Using the Retweet Graph

Retweet graph-based spam detection works as follows. Each participant in the system maintains his own partial[25] list of past messages sent by himself and others. A partial retweet graph is constructed from this dataset, with one vertex per sender and directional edges linking each retweeter to the corresponding retweetees. Denoting the participant's own vertex as the root[26], the remaining participants are classified by two attributes, their distance from the root and the maximum flow from the root to them. Users that are classified as non-spammers are whitelisted.

This approach presents two bootstrapping problems. How does a new user with no recorded history construct a retweet graph and do the messages from a new user that has

[25] Only some messages sent by other will be heard. E.g., in Shout [132] only those message broadcast in the vicinity of the node will be heard and included.

[26] Trusting one's own vertex as non-spammer breaks the otherwise problematic symmetry between the non-spammer and spammer portions of the graph.

never been retweeted ever get seen? For the first question, a user can copy the tweet history from a trusted friend or bootstrap by explicitly whitelisting his friends. For the second, the user can ask his friends to whitelist him, so they can then see and retweet his messages, linking him to the graph. We also anticipate that some (particularly bored) users will choose to view all incoming tweets, retweeting some that are not spam.

The approach relies on the following assumptions.

- Non-spammers retweet spammers much less frequently than non-spammers.
- Spammers only send spam content; they do not bait-and-switch.

The second assumption is, of course, unrealistic. Although it appears to hold in the current dataset, attacks would respond to the deployment of graph-based spam filtering by altering their behavior to manipulate the classification. This method forces spammers to employ some sort of manipulation scheme, increasing the cost of introducing spammer identities (it takes time for the initial good messages to be reshouted) and limiting their total number (practical concerns limit the rate at which users will reshout), but does not provably restrict their advantage. Instead, schemes that are provably resistant to such manipulation, i.e., have bounded attacker advantage [188, 190] could be used, as we discuss Subsection 6.8.5.

For our analysis, we also make the following assumption.

- Spammer retweet spammers and non-spammers in the same fashion that non-spammers retweet non-spammers.

This assumption simplifies the generation of synthetic retweet graphs, but is not necessary for the detection approach. A spammer may choose to behave differently, but doing so simply introduces more structural differences in the retweet graph, making detection even easier.

The following sections analyze the performance of this classification procedure on our 10% sample of the retweet graph and synthetic graphs for parameter sweeps.

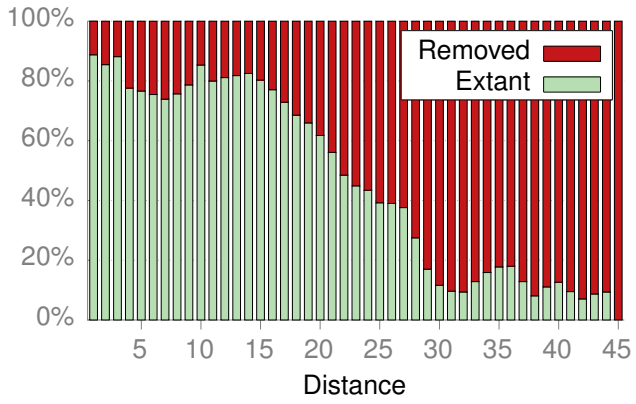


Figure 6.18: Percentage of removed and extant Twitter users as a function of distance from benign users in the retweet graph. Most removed users are spammers, so this graph shows that distance is highly correlated with spammer behavior.

6.8.3 Performance on the Twitter Retweet Graph

We first consider the performance on our 10% sample of the retweet graph. This sample is problematic because most of the paths between non-spammers are not included (90% of edges are missing) but is sufficient to show that the hypothesized differences exist.

We randomly chose 100 source–destination pairs of users who distances in the retweet graph ranged from 1 to 45, for 4500 pairs in total. We obtained ground truth classification for these 9000 users by querying the Twitter API to determine if the account had been removed in the 18 months following the initial collection. Twitter actively seeks out and bans spammers, so the majority of the spammers will have been removed. Some non-spammers will have also deleted their own accounts, so we refer to these categories as *removed* and *extant*. We believe that most removed users were spammers [182].

We consider only the pairs whose source node is extant. Figure 6.18 shows the percentage of destination nodes in each category by the distance from their sources. Clearly, distance in the retweet graph is correlated with spammer tendencies. A classifier over this attribute alone achieves a TPR of 75% with an FPR of 25%.

The second attribute, connectivity, shows no correlation in the 10% sample graph because the majority of edges are missing. Most pairs with between one and ten independent paths in

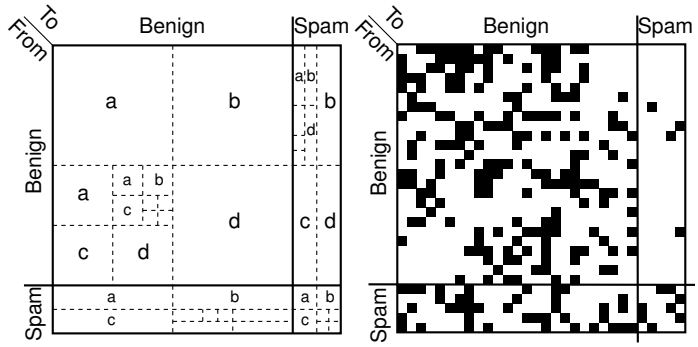


Figure 6.19: Illustration of the modified R-MAT algorithm for generating synthetic retweet graphs and a resulting adjacency matrix. Fewer edges are placed in the benign–spam quadrant to model the lower likelihood of such retweets. Within each quadrant, edges are cascaded in proportion to probabilities a, b, c, and d to generate a scale-free, small-world structure.

the original graph contains only zero or one paths in the sampled graph, making it impossible to distinguish a non-spam node linked by ten paths from a spam node linked by one. Instead, we turn to synthetic retweet graphs to study the performance of the combined classifier.

6.8.4 Performance on Synthetic Retweet Graphs

The analysis in Section 6.6 showed that the retweet graph is scale-free and small-world, enabling the generation of synthetic retweet graphs using the R-MAT (*R*ecursive Matrix), an algorithm designed to generate a variety of such networks [162]. Although metrics like assortativity and clustering are not directly controllable—R-MAT cannot capture the differences between the followers and retweet graphs [27]—it is sufficient for our purposes as we depend only on the connectivity implied by the small-world structure and limited number of incoming edges to spammer nodes.

R-MAT produces scale-free, small-world graphs by treating edge assignment in the adjacency matrix as a two-dimensional binomial cascade. We modify the procedure to generate relatively fewer edges from benign to spammer nodes (B–S) than the other possibilities (B–B, S–S, S–B), modeling the notion that non-spammers rarely retweet spammers.

[27] Unfortunately, this prevents us from comparing retweet-based with follower-based spam detection. A full sample of the retweet graph would be needed.

The modified R-MAT process is illustrated in Figure 6.19. We desire a graph with some number of benign and spammer nodes, some number of non-B–S edges, and a relatively smaller number of B–S edges. The adjacency matrix is divided into four quadrants and the edges split among the B–B, S–S, and S–B quadrants in proportion to their areas. Within each quadrant, the R-MAT algorithm is used to place the edges. For each edge, the sub-quadrant in which to place the edge is chosen according to probabilities a, b, c, and d ($a + b + c + d = 1$). The process recurses until a single cell is selected for the edge. The result of the process for a small graph is shown in Figure 6.19.

The parameters a, b, c, and d are obtained via AutoMAT-fast [162], i.e., fitting the degree distribution of the retweet graph to that of the model. The R-MAT process is essentially a two-dimensional binomial cascade, with the out-edges assigned to the upper and lower halves with probabilities $p \triangleq a + b$ and $1 - p$ and the in-edges assigned to the left and right halves with probabilities $q \triangleq a + c$ and $1 - q$. Letting $N = 2^n$ be the number of nodes and E the number of edges to assign, then the expected number of nodes c_k with out-degree k is

$$c_k = \sum_{i=0}^{n} \binom{n}{i} B\left(k; E, p^{n-i}(1-p)^i\right) \tag{6.21}$$

where $B(k; a, b)$ is the mass function of the binomial distribution $B(a, b)$. The in-edge distribution is computed similarly. Fitting to the retweet graph, we obtain $a = 0.52$, $b = 0.18$, $c = 0.17$, and $d = 0.13$.

We fix the fraction of spam nodes to 10% and assume that spammer retweet behavior mimics that of benign nodes retweeting each other. Differences are not in the attackers' interest, as they would enable additional classification methods.

The performance of the classifier is primarily affected by two metrics—the fraction of possible B–B edges that are present and the number of B–S edges per spammer vertex—so we conduct parameter sweeps of these values.

If the B–B edge density is too low, many benign pairs will not be connected and the false

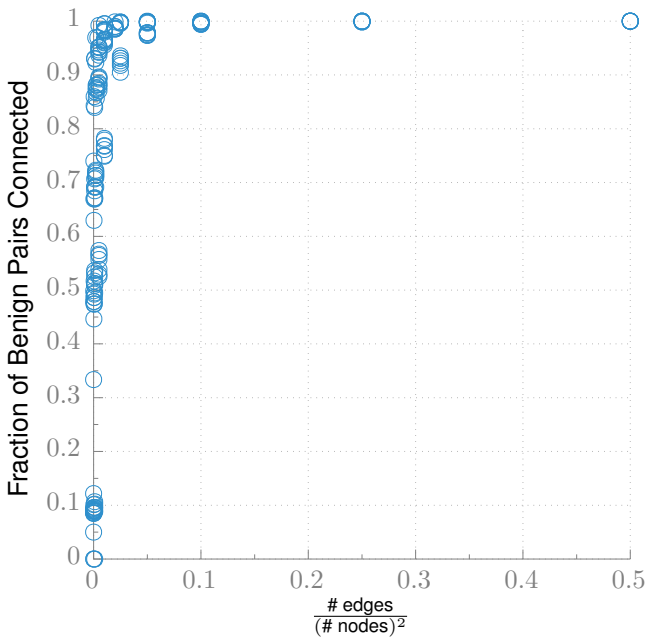


Figure 6.20: Connectivity of benign pairs as a function of the benign edge density. Above 5%, almost all pairs are connected. We expect that density does not grow with network size, so this limits the network size for which the false positive rate is acceptable. For large networks, the technique will only work within clusters.

positive rate will be high. Figure 6.20 plots this density against the fraction of benign pairs that are connected for a variety of network sizes. Above 5%, most pairs are connected and above 10%, essentially all pairs are connected. We expect the number of edges in a retweet graph to (above some point) grow linearly in the number of users, so this relationship places a limit on the network size for which the technique is usable. For larger networks (e.g., the world population), the technique will only work within clusters for which the edge density is high enough—users outside of one's own cluster will be identified as spammers. For example, the average out-degree of Twitter, 75, would support 25000 participants. However, social relationships are clustered, so this limitation should rarely be an issue in practice.[28] In a network like Shout [132], the effective community size is already limited by geography.

Figure 6.21 shows the classification performance. We use the J48 decision tree classifier

[28]This limitation does prevent the discovery of content from outside of one's own group, possible with centralized Twitter today. Content can still traverse two groups if seen and retweeted by a member of both.

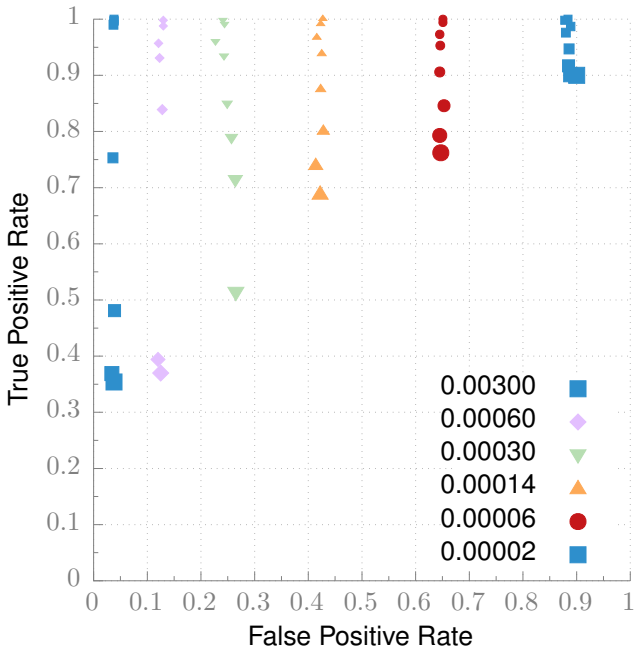


Figure 6.21: Performance of J48 classifier over distance and connectivity attributes in the synthetic graphs. The benign edge density (marker symbol and color) range from 0.00002 to 0.003 and the number of B–S edges per spammer node (marker size) ranges from 0.01 to 1. Each marker is a single point on the resulting ROC curve.

over both the distance and connectivity (max-flow) attributes, using 10-fold cross validation. We sweep both the benign edge density (marker symbol and color) from 0.0002 to 0.003 and the number of B–S edges per spammer (marker size) from 0.01 to 1. To reduce clutter, a *single* point[29] from each resulting ROC curve is plotted. Two trends are immediately clear. Decreasing the benign edge density increases the FPR, but an FPR below 5% requires just a 0.3% edge density. Increasing the B–S rate (number of B–S edges per spammer node) decreases the true positive rate. If less than one-tenth of spammers are retweeted by benign nodes, the TPR is universally above 98%. The sensitivity to B–S rate increases with edge density because the spammer nodes are more interconnected (we hold the S–B and S–S densities equal to the B–B density).

This performance—98% TPR and <5% FPR—is consistent with the results observed on

[29]The selected points are generally near the knees of the curves, but within a class are intentionally chosen to have similar FPRs.

the Twitter followers graph by Song, Lee, and Kim [165] and substantially better than the 70% TPR and 4% FPR observed by Benevenuto et al. [164] for attribute-based classification. The false positives are due to the low edge density of in the cluster of benign nodes. Each user only retweets an average of 75 other users, so as the node count increases, that cluster becomes increasingly disconnected. The false negatives are due to benign nodes occasionally retweeting spammers. A few spammers are (randomly) retweeted by enough users to become connected to the graph. These retweets are rare and essentially random though (e.g., an accidental press of the retweet button) and thus the false negative rate remains low.

In summary, Figure 6.18 shows that the inter-node distance in the retweet graph is highly correlated with being a spammer, enabling detection. Simulations on the synthetic graphs show that inter-node distance and inter-node max flow can identify spammers with greater than 98% TPR and less than 5% FPR when fewer than one-tenth of spammers are retweeted and at least 0.3% of possible edges between benign nodes are present. For a community-sized network of 25000 participants, this implies an average node degree of 75, i.e., that of the Twitter retweet graph. For larger networks, the classification works best within smaller sub-clusters where the edge density is higher.

6.8.5 Discussion of Provably Manipulation-Resistant Schemes

We have shown only that the retweet graph-based spam filtering method works against current spammer behavior. Were the technique deployed, spammers could alter their behavior to manipulate the structure of the graph, e.g., by initially tweeting some non-spam content to gain retweets. Detection techniques that are provably resistant to such manipulation would be preferred. In this section, we briefly summarize how two such techniques, developed in the context of recommender systems but generally-applicable, can be applied to spam detection. Full evaluation is beyond our scope and is left for future work.

We first consider the *influence limiter* [188], an overlay scheme to make any recommender scheme provably manipulation-resistant. Here, the goal is to generate recommenda-

tions (over some domain of items) for a user (the target) using the ratings provided by other users (the raters). Dishonest (and possibly Sybil) raters can submit false ratings designed to manipulate the generated recommendations. The influence limiter computes an influence for each rater based on the (predictive) accuracy of his ratings. Ratings that improve the recommendation accuracy (measured by how much better the prediction matches the target's *future* ratings) increase influence and vice versa. When generating a recommendation, ratings are weighted by their respective rater's influence. New users are bootstrapped by starting them with a small amount of initial influence. This formulation maps directly to retweet-based spammer detection. The tweeters are the raters and the contents being tweeted are the items. A tweet serves as a positive rating for that content. A retweet by a target indicates agreement with that rating. This scheme is (n,c)-robust (the expected damage by an attacker controlling n identities is upper bounded by c) with information loss (honest ratings that are under-weighted during recommendation) of the some order of magnitude as a (possibly loose) provable lower bound [189].

The influence limiter does not chain the rater reputations transitively, e.g., a target A assigning high influence to rater B who assigned high influence to rater C does not directly result in A assigning high influence to C. Like the retweet graph-based classifier, we can consider transitive trust relationships, but in a manipulation-resistant fashion [190]. In this model, each participant associates a *trust balance* with each other participant. These trust balances are used to decide whether a participant should engage in an interaction (e.g., do I have enough, possibly transitive, trust in that user to "cover" the risk of the interaction). In our case, a user (the principal) reading a tweet from another user (the agent) is a transaction. An (honest) principal indicates that the transaction was successful (i.e., the tweet was useful) by retweeting it. Intuitively, the trust balances of any participants involved in an interaction (i.e., including those used for transitive trust) should be increased (decreased) after successful (unsuccessful) interactions between participants. The choice of trust update protocol determines the manipulation-resistance, as attackers may lie about success (i.e.,

retweet spam). Specifically, we desire that the incorporation of transitive trust not give the attacker (who can control many participants) any more advantage than he would have if only direct trust was used. One formalization of this property is called *sum-sybilproofness* and is provided by the *hedged-transitive* update protocol [190]. Unfortunately, it is also known that any sum-sybilproof protocol will reject some interactions that would be allowed by a protocol employing direct trust only, limiting the overall usefulness of indirect trust.

Spam detection based on these techniques (i.e., a manipulation-resistant recommender system [188, 189] and sybil-sumproof incorporation of transitive trust [190]) offer a significant improvement over the retweet-graph based approach. Their manipulation-resistance is provably quantifiable (and tight against theoretical bounds), no matter how clever the attacker. Future efforts towards implementing spam detection in Shout should incorporate them.

6.9 Derivation of the EM Method

Using the same notation as Subsection 6.2.5, the likelihood to maximize is

$$\begin{aligned}
\mathcal{L}_C(\phi|f,g) &= \log p(f,g|\phi) && (6.22)\\
&\propto \log p(f|\phi) && (6.23)\\
&\propto \log \prod_{1\le j\le i} \left(\phi_i c_{i,j}\right)^{f_{i,j}} && (6.24)\\
&= \sum_{1\le j\le i} f_{i,j} \log\left(\phi_i c_{i,j}\right). && (6.25)
\end{aligned}$$

The expected likelihood under an estimate ϕ^k is

$$\begin{aligned}
\mathcal{Q}(\phi,\phi^{(k)}) &\triangleq \mathsf{E}_{f|g,\phi^{(k)}}\left[\mathcal{L}_C(\phi|f,g)\right] && (6.26)\\
&= \sum_{1\le j\le i} \mathsf{E}_{\phi^{(k)}}\left[f_{i,j}|g\right] \log\left(\phi_i c_{i,j}\right) && (6.27)
\end{aligned}$$

and the iterative maximization step is

$$\phi^{(k+1)} \triangleq \arg\max_{\phi} \mathcal{Q}(\phi, \phi^{(k)}). \tag{6.28}$$

The maximum is computed under the constraint $\sum_{1\leq i} \phi_i = 1$ using Lagrangian multipliers. Defining the Lagrangian

$$L(\phi, \lambda) \triangleq \sum_{1\leq j\leq i} \mathsf{E}_{\phi^{(k)}}\left[f_{i,j}|g\right] \log\left(\phi_i c_{i,j}\right) + \lambda(1 - \sum_{1\leq i} \phi_i), \tag{6.29}$$

the associated partial derivatives are

$$\frac{\partial L}{\partial \phi_i} = \frac{\mathsf{E}_{\phi^{(k)}}\left[f_{i,j}|g\right]}{\phi_i} - \lambda, \text{ and} \tag{6.30}$$

$$\frac{\partial L}{\partial \lambda} = 1 - \sum_{1\leq i} \phi_i \tag{6.31}$$

Solving for

$$\phi_i = \frac{\mathsf{E}_{\phi^{(k)}}\left[f_i|g\right]}{\sum_{1\leq l} \mathsf{E}_{\phi^{(k)}}\left[f_l|g\right]} \tag{6.32}$$

and defining

$$\gamma \triangleq \sum_{1\leq l} \mathsf{E}_{\phi^{(k)}}\left[f_l|g\right] = \sum_{1\leq l} g_l \tag{6.33}$$

yields

$$\phi_i^{(k+1)} = \frac{\mathsf{E}_{\phi^{(k)}}\left[f_i|g\right]}{\gamma} \tag{6.34}$$

$$= \frac{\phi_i^{(k)}}{\gamma} \sum_j \frac{c_{i,j} g_j}{\sum_{1\leq l} \phi_l^{(k)} c_{l,j}}. \tag{6.35}$$

or in matrix form (for fast implementation on a computer)

$$\phi^{(k+1)} = \frac{1}{\gamma} \times \phi^{(k)} \times C \cdot \frac{g}{C^\top \cdot \phi^{(k)}}. \tag{6.36}$$

The original frequencies can be expressed as

$$\hat{f}_i = \gamma \phi_i \frac{1}{1 - B_{0.1}(i, 0)}. \tag{6.37}$$

6.10 Conclusion

We have presented an initial characterization of aggregate user behavior, describing the distributions of lifetime contributions, tweet rates, and inter-tweet durations. These behaviors are thought to be common across communication platforms, but our results differ from prior analysis, suggesting future study to determine the true extent of the similarities. Our retweet graph analysis revealed structural differences from the followers graph that are more consistent with real world social networks. Explaining the underlying causes of the observed differences—we conjecture that retweets more closely mirror real-world relationships and trust—is an open problem. Finally, we developed a method for detecting spammers via their low connectivity in the retweet graph.

CHAPTER 7

Conclusion

This thesis has advocated the development of non-hierarchical networks to combat censorship and surveillance in communication networks. Continued work is needed to ready such systems for everyday use, but this thesis has taken the following steps.

- *Private, reprisal-resistant communication for friends and family:* We proposed Whisper, a MANET architecture that uses a novel routing scheme based on the predictability of human motion to increase scalability. Privacy and anonymity are provided by a novel onion-routing variant that does not require a priori selection of potential onion routers. The Mason test was developed to enable the use of random nodes from those encountered during daily travels for onion routing, ensuring that the proportion of selected onion routers that are attackers is limited by the proportion of physical, participating nodes owned by the attackers.

- *Censorship-resistant public microblogging:* We proposed Shout, a MANET microblogging architecture that uses geographic proximity and manual human action to disseminate messages in a non-hierarchical fashion. To combate spam, we developed a novel spammer detection technique based on the intuition that messages from spammers will be repeated and forwarded less frequently that those from non-spammers. We developed analytical models of user behavior in Twitter to enable simulation-based study and optimization of Shout-like microblogging systems. We used our characteri-

zation of the retweet graph to generate random reshout graphs suitable for studying the classification performance of our spammer detection technique.

The results described in this thesis represent early steps in the development of non-hierarchical networks. Current smartphones have extremely limited battery capacities, limiting the potential for Whisper, which relies on other devices to forward messages. Both Whisper and Shout require a critical mass of users before messages can propagate far enough to be useful. Centralized services will often be more convenient. The public must care about privacy before they will tolerate the inevitable inconveniences of decentralized and non-hierarchical alternatives. I hope that once privacy has become a first-order requirement for the public, the methods presented here are useful in the development of privacy-preserving communication platforms.

BIBLIOGRAPHY

[1] D. R. Bild, Y. Liu, R. P. Dick, Z. M. Mao, and D. Wallach, "Using predictable mobility patterns to support scalable and secure MANETs of handheld devices," in *Proc. Int. Wkshp. on Mobility in the Evolving Internet Architecture*, June 2011, pp. 13–18.

[2] Z. Wilcox-O'Hearn, "Names: Decentralized, secure, human-meaningful: Choose two," https://zooko.com/uri/URI:DIR2-RO:d23ekhh2b4xashf53ycrfoynkq:y4vpazbrt2beddyhgwcch4sduhnmmefdotlyelojxg4tyzllhb4a/distnames.html.

[3] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong, "On the Levy-walk nature of human mobility," in *Proc. Int. Conf. Computer Communications*, Apr. 2008, pp. 924–932.

[4] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, "SLAW: a mobility model for human walks," in *Proc. Int. Conf. Computer Communications*, Apr. 2009, pp. 855–863.

[5] L. Xiao, L. J. Greenstein, N. B. Mandayam, and W. Trappe, "Channel-based detection of Sybil attacks in wireless networks," *IEEE Trans. Information Forensics and Security*, vol. 4, no. 3, pp. 492–503, Sept. 2009.

[6] D. B. Faria and D. R. Cheriton, "Detecting identity-based attacks in wireless networks using signalprints," in *Proc. Wkshp. Wireless Security*, Sept. 2006, pp. 43–52.

[7] A.-L. Barabási, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, and T. Vicsek, "Evolution of the social network of scientific collaborations," *Physica A: Statistical Mechanics and its Applications*, vol. 311, no. 3–4, pp. 590–614, Aug. 2002.

[8] S. N. Dorogovtsev and J. F. F. Mendes, "Language as an evolving word web," *Proc. Royal Society London B*, vol. 268, no. 1485, pp. 2603–2606, Dec. 2001.

[9] J. Zittrain and B. Edelman, "Internet filtering in china," *IEEE Internet Computing*, vol. 7, no. 2, pp. 70–77, Mar. 2003.

[10] T. Zhu, D. Phipps, A. Pridgen, J. R. Crandall, and D. S. Wallach, "The velocity of censorship: High-fidelity detection of microblog post deletions," in *Proc. USENIX Security Symp.*, Aug. 2013, pp. 227–240.

[11] T. Branigan, "China blocks Twitter, Flickr, and Hotmail ahead of Tiananmen anniversary," *The Guardian*, June 2 2009.

[12] H. Noman and J. C. York, "West censoring east: The user of western technologies by middle east censors, 2010–2011," OpenNet Initiative, Tech. Rep., Mar. 2011.

[13] Reports Without Borders, "Enemies of the internet report 2012," pp. 1–71, Mar. 2012.

[14] E. Schonfeld, "Twitter is blocked in Egypt amidst rising protests," *TechCrunch*, Jan. 25 2011, http://www.techcrunch.com/2011/01/25/twitter-blocked-egypt.

[15] OpenNet Initiative, "Internet filtering in Tunisia," 2009, http://opennet.net/research/profiles/tunisia.

[16] N. Anderson, "Tweeting tyrants out of Tunisia: Global Internet at its best," *Wired.com*, Jan. 14 2011, http://www.wired.com/threatlevel/2011/01/tunisia.

[17] The Wall Street Journal, "Egypt communications cut ahead of further protests," Jan. 28 2011, http://online.wsj.com/article/BT-CO-20110128-706943.html.

[18] J. Cowie, "Egypt leaves the Internet," *Renesys Blog*, Jan. 27 2011, http://www.webcitation.org/query?url=www.renesys.com/blog/2011/01/egypt-leaves-the-internet.shtml.

[19] C. to Protect Journalists, "Committee to protect journalists 2008 prison census: Online and in jail," Dec. 2008, http://www.cpj.org/imprisoned/cpjs-2008-census-online-journalists-now-jailed-mor.php.

[20] J. Goldsmith and T. Wu, *Who Controls the Internet?* Oxford University Press, 2006.

[21] R. Marquand, "The 'mouse' that caused an uproar," Nov. 2003.

[22] J. Risen and E. Lichtblau, "Bush lets U.S. spy on callers without courts," *N.Y. Times*, Dec. 16 2005.

[23] L. Davidson and A. J. O'Donoghue, "Utah will host new $1.9 billion NSA spy center," *Deseret News*, Jul. 3 2009.

[24] M. D. Laplante, "Spies like us: NSA to build huge facility in Utah," *The Salt Lake Tribune*, Jul. 1 2009.

[25] H. Hoogstraaten, et al., "Black tulip: Report of the investigation into the DigiNotar certificate authority breach," Fox-IT, Tech. Rep., Aug. 2012.

[26] A. Whitten and J. D. Tygar, "Why Johnny can't encrypt: a usability evaluation of PGP 5.0," in *Proc. USENIX Security Symp.*, Aug. 1999, pp. 169–184.

[27] B. Schneier, *Applied Cryptography*. John Wiley & Sons, 1996.

[28] E. Rescorla, *SSL and TLS: Designing and Building Secure Systems*. Addison-Wesley Professional, 2000.

[29] P. R. Zimmermann, *The official PGP user's guide*. MIT Press, 1995.

[30] N. Borisov, I. Goldberg, and E. Brewer, "Off-the-record communication, or, why not to use PGP," in *Proc. Wkshp. Privacy in the Electronic Society*, Oct. 2004, pp. 77–84.

[31] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–88, Feb. 1981.

[32] G. Danezis, R. Dingledine, and N. Mathewson, "Mixminion: Design of a type III anonymous remailer protocol," in *Proc. Symp. Security and Privacy*, May 2003, pp. 2–15.

[33] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: the second-generation onion router," in *Proc. USENIX Security Symp.*, Aug. 2004, p. 21.

[34] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman, "Telex: Anticensorship in the network infrastructure," in *Proc. USENIX Security Symp.*, Aug. 2011, pp. 1–15.

[35] P. Zimmermann, A. Johnston, and J. Callas, "ZRTP: Media Path Key Agreement for Unicast Secure RTP," RFC 6189 (Informational), Internet Engineering Task Force, Apr. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6189.txt

[36] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Information Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[37] J. Wu, Y. Zhang, Z. M. Mao, and K. Shin, "Internet routing resilience to failures: Analysis and implications," in *Proc. Int. Conf. Emerging Networking Experiments & Technologies*, Dec. 2007, pp. 1–12.

[38] F. Xue and P. Kumar, *Scaling Laws for Ad Hoc Wireless Networks: An Information Theoretic Approach*. NOW Publishers, 2006.

[39] R. Pike, "More unrest in the Middle East results in Internet disruptions," *TechieInsider.com*, Feb. 19 2011, http://www.webcitation.org/query?url=www.techieinsider.com/news/6485&date=2011-03-17.

[40] B. Karp and H. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proc. Int. Conf. Mobile Computing and Networking*, Aug. 2000, pp. 243–254.

[41] R. Barr, Z. J. Haas, and R. van Renesse, "Scalable wireless ad hoc network simulation," in *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, J. Wu, Ed. CRC Press, 2005, ch. 19, pp. 297–311.

[42] C. Bettstetter, "On the connectivity of ad hoc networks," *The Computer Journal*, vol. 47, no. 4, pp. 432–447, 2004.

[43] C. Cortes and D. Pregibon, "Signature-based methods for data streams," *Data Mining and Knowledge Discovery*, vol. 5, no. 3, pp. 167–182, July 2001.

[44] A. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Computing*, vol. 2, pp. 46–55, Jan. 2003.

[45] S. M. Das, H. Pucha, and Y. C. Hu, "Performance comparison of scalable location services for geographic ad hoc routing," in *Proc. Int. Conf. Computer Communications*, Mar. 2005, pp. 1228–1239.

[46] M. C. González, C. A. Hidalgo, and A.-L. Barabási, "Understanding individual human mobility patterns," *Nature*, vol. 453, pp. 778–782, June 2008.

[47] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Proc. Int. Multi-Topic Conf.*, Dec. 2001, pp. 62–68.

[48] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 1, pp. 1–22, Jan. 2004.

[49] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proc. Wkshp. on Mobile Computing Systems and Applications*, Feb. 1999, pp. 90–100.

[50] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, vol. 353, pp. 153–181, 1996.

[51] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing," in *Proc. Int. Conf. Mobile Computing and Networking*, Aug. 2000.

[52] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human motion," *Science*, vol. 327, pp. 1018–2021, Feb. 2010.

[53] I. Burbey and T. L. Martin, "Predicting future locations using prediction-by-partial-match," in *Proc. Int. Wkshp. Mobile Entity Localization and Tracking in GPS-less Environments*, Sept. 2008, pp. 1–6.

[54] M. McNett and G. M. Voelker, "Access and mobility of wireless PDA users," *Mobile Computing Communications Review*, vol. 9, no. 2, pp. 40–55, Apr. 2005. [Online]. Available: http://sysnet.ucsd.edu/wtd/

[55] D. J. Aldous and W. S. Kendall, "Short-length routes in low-cost networks via Poisson line patterns," *Advances in Applied Probability*, vol. 40, no. 1, pp. 1–21, Mar. 2008.

[56] B. Schneier, "Why 'anonymous data' sometimes isn't," *Wired.com*, Dec. 13 2007, http://www.webcitation.org/query?url=www.wired.com/politics/security/commentary/securitymatters/2007/12/securitymatters_1213&date=2011-04-27.

[57] J. Douceur, "The Sybil attack," in *Proc. Int. Wkshp. Peer-to-Peer Systems*, Mar. 2002, pp. 251–260.

[58] @Twitter, https://twitter.com/twitter/status/281051652235087872.

[59] D. Etherington, "Twitter passes 200M monthly active users, a 42% increase over 9 months," TechCrunch, Dec. 18 2012, http://techcrunch.com/2012/12/18/twitter-passes-200m-monthly-active-users-a-42-increase-over-9-months/.

[60] S. Millward, "Sina reveals Q3 financials, announces Weibo has passed 400 million registered users," Tech In Asia, Nov. 16 2012, http://www.techinasia.com/sina-weibo-400-million-registered-users/.

[61] P. N. Howard, A. Duffy, D. Freelon, M. Hussain, W. Mari, and M. Mazaid, "Opening closed regimes: What was the role of social media during the Arab Spring," Project on Information Technology & Political Islam, Sept. 2011.

[62] A. Harjani, "This could sparks China's Arab Spring," CNBC, Mar. 7 2013, http://www.cnbc.com/id/100535405.

[63] "Dynaweb proxy," http://www.dit-inc.us/dynaweb.

[64] "Ultrasurf proxy," https://ultrasurf.us/.

[65] "Speak to tweet," https://twitter.com/speak2tweet.

[66] R. Faris, H. Roberts, and S. Wang, "China's green dam," OpenNet Initiative, 2009.

[67] S. Wolchok, R. Yao, and J. A. Halderman, "Analysis of the Green Dam censorware system," Computer Science and Engineering Division, University of Michigan, Tech. Rep. 18, 2009.

[68] U. of Michigan Emergency Management Team, Personal Communication.

[69] D. R. Sandler and D. S. Wallach, "Birds of a FETHR: Open, decentralized micropublishing," in *Proc. Int. Wkshp. Peer-to-Peer Systems*, Apr. 2009, pp. 1–6.

[70] T. Xu, Y. Chen, J. Zhao, and X. Fu, "Cuckoo: Towards decentralized, socio-aware online microblogging services and data measurements," in *Proc. HotPlanet Wkshp.*, June 2010, pp. 1–6.

[71] P. St Juste, D. Wolinsky, P. O. Boykin, and R. J. Figueiredo, "Litter: A lightweight peer-to-peer microblogging service," in *Proc. Int. Conf. Privacy, Security, Risk and Trust*, Oct. 2011, pp. 900–903.

[72] "Wi-fi direct," www.wi-fi.org/discover-and-learn/wi-fi-direct.

[73] M. Stiegler, "An introduction to petname systems," Feb. 2005, http://www.skyhunter.com/marcs/petnames/IntroPetNames.html.

[74] A. Swartz, "Squaring the triangle: Secure, decentralized, human-readable names," Jan. 2011, http://www.aaronsw.com/weblog/squarezooko.

[75] "Namecoin," http://namecoin.info.

[76] "Bitcoin," http://bitcoin.org.

[77] NIST, "Recommendation for key management—part 1: General (revision 3)," July 2012, http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf.

[78] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on twitter," in *Proc. Collaboration, Electronic Messaging, Anti-Abuse and Spam Conf.*, July 2011, pp. 1–9.

[79] O. Fletcher, "Years on, China pushes WAPI in mobile phones," CIO, May 2009.

[80] C. Shu, "Proposed Chinese law may force Sina Weibo to implement real-name registration," TechCrunch, Dec. 2012, http://techcrunch.com/2012/12/23/proposed-chinese-law-may-force-sina-weibo-to-implement-real-name-registration/.

[81] A. Abuy, "Twitter users in Saudi Arabia maybe required to use their real name," Kabayan Tech, Mar. 2013, http://kabayantech.com/2013/03/twiter-users-in-saudi-arabia-maybe-required-to-use-their-real-name/.

[82] "Market share: Mobile communication devices by region and country, 3Q11," Gartner, Nov. 2011.

[83] "Alljoyn," http://www.alljoyn.org.

[84] A. J. Nicholson, S. Wolchok, and B. D. Noble, "Juggler: Virtual networks for fun and profit," *IEEE Trans. Mobile Computing*, vol. 9, no. 1, pp. 31–43, Jan. 2010.

[85] "Juggler: An open-source virtual link layer for Linux," http://www.eecs.umich.edu/∼tonynich/juggler/.

[86] R. Chandra, P. Bahl, and P. Bahl, "MultiNet: connecting to multiple IEEE 802.11 networks using a single wireless card," in *Proc. Int. Conf. Computer Communications*, vol. 2, Mar. 2004, pp. 882–893.

[87] S. Kandula, K. C.-J. Lin, T. Badirkhanli, and D. Katabi, "FatVAP: Aggregating AP backhaul capacity to maximize throughput," in *Proc. USENIX Symp. Networked Systems Design and Implementation*, Apr. 2008.

[88] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *Proc. Int. Conf. Computer Communications*, Mar. 2003, pp. 1312–1321.

[89] D. R. Choffnes and F. E. Bustamante, "An integrated mobility and traffic model for vehicular wireless networks," in *Proc. Int. Wkshp. Vehicular Ad Hoc Networks*, Sept. 2005, pp. 69–78.

[90] C. Boldrini and A. Passarella, "HCMM: modelling spatial and temporal properties of human mobility driven by users' social relationships," *Computer Communication*, vol. 33, no. 9, pp. 1056–1074, June 2010.

[91] Y.-C. Chen, E. M. Nahum, R. J. Gibbens, D. Towsley, and Y. sup Lim, "Characterizing 4G and 3G networks: Supporting mobility with multi-path TCP," School of Computer Science, University of Massachusetts Amherst, Tech. Rep. 22, 2012.

[92] C. Benvenuti, *Understanding Linux Network Internals*, 1st ed. O'Reilly Media, Jan. 2006.

[93] "Project voldemort," http://www.project-voldemort.com/voldemort/.

[94] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Dynamo: Amazon's highly-available key-value store," in *Proc. Symp. Operating Systems Principles*, Oct. 2007, pp. 205–220.

[95] R. Maheshwari, S. Jain, and S. R. Das, "A measurement study of interference modeling and scheduling in low-power wireless networks," in *Proc. Conf. Embedded Network Sensor Systems*, Nov. 2008, pp. 1–14.

[96] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE rap: Social-based forwarding in delay tolerant networks," *IEEE Trans. Mobile Computing*, vol. 10, no. 11, pp. 1576–1589, Nov. 2011.

[97] Y. Xiang, L. S. Bai, R. Piedrahita, R. P. Dick, Q. Lv, M. P. Hannigan, and L. Shang, "Collaborative calibration and sensor placement for mobile sensor networks," in *Proc. Int. Conf. Information Processing in Sensor Networks*, Apr. 2012, pp. 73–84.

[98] P. Gardner-Stephen, "The Serval project: Practical wireless ad-hoc mobile telecommunications," Flinders University, Adelaide, South Australia, Tech. Rep., Aug. 2011.

[99] J. Newsome, E. Shi, D. Song, and A. Perrig, "The Sybil attack in sensor networks: Analysis & defenses," in *Proc. Int. Conf. Information Processing in Sensor Networks*, Apr. 2004, pp. 259–268.

[100] B. N. Levine, C. Shields, and N. B. Margolin, "A survey of solutions to the Sybil attack," Department of Computer Science, University of Massachusetts Amherst, Amherst, MA, Tech. Rep., Oct. 2006.

[101] H. Zhou, M. Mutka, and L. Ni, "Multiple-key cryptography-based distributed certificate authority in mobile ad-hoc networks," in *Proc. Global Telecommunications Conf.*, Nov. 2005.

[102] M. Ramkumar and N. Memon, "An efficient key predistribution scheme for ad hoc network security," *IEEE J. Selected Areas in Communications*, vol. 23, pp. 611–621, Mar. 2005.

[103] N. Borisov, "Computational puzzles as Sybil defenses," in *Proc. Int. Conf. Peer-to-Peer Computing*, Sept. 2006, pp. 171–176.

[104] F. Li, P. Mittal, M. Caesar, and N. Borisov, "SybilControl: Practical Sybil defense with computational puzzles," in *Proc. Wkshp. Scalable Trusted Computing*, Oct. 2012.

[105] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: defending against Sybil attacks via social networks," in *Proc. ACM SIGCOMM Computer Communication Review*, Sept. 2006, pp. 267–278.

[106] H. Yu, P. Gibbons, M. Kaminsky, and F. Xiao, "SybilLimit: A near-optimal social network defense against Sybil attacks," in *Proc. Symp. Security and Privacy*, May 2008, pp. 3–17.

[107] T. S. Rappaport, *Wireless Communications: Principles & Practice*. Prentice-Hall, NJ, 2002.

[108] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki, "Practical robust localization over large-scale 802.11 wireless networks," in *Proc. Int. Conf. Mobile Computing and Networking*, Sept. 2004, pp. 70–84.

[109] M. Demirbas and Y. Song, "An RSSI-based scheme for Sybil attack detection in wireless sensor networks," in *Proc. Int. Symp. on a World of Wireless, Mobile, and Multimedia*, June 2006, pp. 564–570.

[110] Z. Li, W. Xu, R. Miller, and W. Trappe, "Securing wireless systems via lower layer enforcements," in *Proc. Wkshp. Wireless Security*, Sept. 2006, pp. 33–42.

[111] Q. Li and W. Trappe, "Detecting spoofing and anomalous traffic in wireless networks via forge-resistant relationships," *IEEE Trans. Information Forensics and Security*, vol. 2, no. 4, pp. 793–803, Dec. 2007.

[112] Y. Chen, J. Yang, W. Trappe, and R. P. Martin, "Detecting and localizing identity-based attacks in wireless and sensor networks," *IEEE Trans. Vehicular Technology*, vol. 5, no. 5, pp. 2418–2434, June 2010.

[113] T. Suen and A. Yasinsac, "Peer identification in wireless and sensor networks using signal properties," in *Proc. Int. Conf. Mobile Adhoc and Sensor Systems*, Nov. 2005, pp. 826–833.

[114] S. Lv, X. Wang, X. Zhao, and X. Zhou, "Detecting the Sybil attack coorperatively in wireless sensor networks," in *Proc. Int. Conf. Computational Intelligence and Security*, Dec. 2008, pp. 442–446.

[115] S. Abbas, M. Merabti, and D. Llewellyn-Jones, "Signal strength based Sybil attack detection in wireless ad hoc networks," in *Proc. Int. Conf. Developments in eSytems Engineering*, Dec. 2009, pp. 22–33.

[116] M. S. Bouassida, G. Guette, M. Shawky, and B. Ducourthial, "Sybil nodes detection basedon received strength variations within VANET," *Int. J. Network Security*, vol. 9, no. 1, pp. 22–33, July 2009.

[117] D. Gesbert, M. Shafi, D. Shiu, P. J. Smith, and A. Naguib, "From theory to practice: An overview of MIMO space–time coded wireless systems," *IEEE J. Selected Areas in Communications*, vol. 21, no. 3, pp. 281–302, Apr. 2003.

[118] Y. Liu, D. R. Bild, and R. P. Dick, "Extending channel comparison based Sybil detection to MIMO systems," Dept. of Electrical Engineering and Computer Science, University of Michigan, Tech. Rep., http://www.davidbild.org/publications/liu13dec.pdf.

[119] H. Hashemi, D. Lee, and D. Ehman, "Statistical modeling of the indoor radio propagation channel – part II," in *Proc. Vehicular Technology Conf.*, May 1992, pp. 839–843.

[120] T. S. Rappaport, S. Y. Seidel, and K. Takamizawa, "Statistical channel impulse response models for factory and open plan building radio communication system design," *IEEE Trans. on Communications*, vol. 39, no. 5, pp. 794–806, May 1991.

[121] "Reaction time statistics," http://www.humanbenchmark.com/tests/reactiontime/stats.php.

[122] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter, a social network or a news media?" in *Proc. Int. World Wide Web Conf.*, Apr. 2010, pp. 591–600. [Online]. Available: http://an.kaist.ac.kr/traces/WWW2010.html

[123] C. A. Bliss, I. M. Kloumann, K. D. Harrison, C. M. Danforth, and P. S. Dodds, "Twitter reciprocal reply networks exhibit assortativity with respect to happiness," *J. Computational Science*, vol. 3, pp. 388–397, Sept. 2012.

[124] A. R. M. Teutle, "Twitter: Network properties analysis," in *Proc. Int. Conf. Electronics, Communications, and Computer*, Feb. 2010, pp. 180–186.

[125] M. Gabielkov and A. Legout, "The complete picture of the Twitter social graph," in *Proc. Int. Conf. Emerging Networking Experiments and Technologies Student Wkshp.*, Dec. 2012, pp. 19–20.

[126] S. Ghosh, A. Srivastava, and N. Ganguly, "Effects of a soft cut-off on node-degree in the Twitter social network," *Computer Communications*, vol. 35, no. 7, pp. 784–795, Apr. 2012.

[127] B. Suh, L. Hong, P. Pirolli, and E. H. Chi, "Want to be retweeted? Large scale analytics on factors impacting retweet in Twitter network," in *Proc. Int. Conf. Social Computing*, Aug. 2010, pp. 177–184.

[128] A. Java, X. Song, T. Finin, and B. Tseng, "Why we Twitter: Understanding microblogging usage and communitites," in *Proc. Wkshp. Web Mining and Social Network Analysis*, Aug. 2007, pp. 56–65.

[129] S. Wu, J. M. Hofman, W. A. Mason, and D. J. Watts, "Who says what to whom on Twitter," in *Proc. Int. World Wide Web Conf.*, Mar. 2011, pp. 705–714.

[130] G. Lotan, E. Graeff, M. Ananny, D. Gaffney, I. Pearce, and D. Boyd, "The revolutions were tweeted: Information flows during the 2011 Tunisian and Egyptian revolutions," *Int. J. Communication*, vol. 5, pp. 1375–1405, 2011.

[131] W. Galuba, K. Aberer, D. Chakraborty, Z. Despotovic, and W. Kellerer, "Outtweeting the Twitterers - predicting information cascades in microblogs," in *Proc. Wkshp. Online Social Networks*, June 2010.

[132] "Shout: Censorship-resistant microblogging," 2013. [Online]. Available: http://whispercomm.org/shout/

[133] M. Freitas, "Twister: Peer-to-peer microblogging," 2013. [Online]. Available: http://twister.net.co/

[134] D. M. Wilkinson, "Strong regularities in online peer production," in *Proc. Conf. Electronic Commerce*, July 2008, pp. 302–309.

[135] M. Seshadri, S. Machiraju, A. Sridharan, J. Bolot, C. Faloutsos, and J. Leskovec, "Mobile call graphs: Beyond power-law and lognormal distributions," in *Proc. Int. Conf. Knowledge Discovery and Data Mining*, Aug. 2008, pp. 596–604.

[136] J. Candia, M. C. González, P. Wang, T. Schoenharl, G. Madey, and A.-L. Barabási, "Uncovering individual and collective human dynamics from mobile phone records," *J. of Physics A: Mathematical and Theoretical*, vol. 41, no. 22, p. 224015, June 2008.

[137] A. Watters, "How recent changes to Twitter's terms of service might hurt academic research," Mar. 2011, http://readwrite.com/2011/03/03/how_recent_changes_to_twitters_terms_of_service_mi. [Online]. Available: http://webcitation.org/6MgAFaaMi

[138] L. A. Goodman, "Snowball sampling," *Annals Mathematical Statistics*, vol. 32, no. 1, pp. 148–170, Mar. 1961.

[139] S. H. Lee, P.-J. Kim, and H. Jeong, "Statistical properties of sampled networks," *APS Physical Review E*, vol. 73, no. 1, pp. 016 102:1–7, Jan. 2006.

[140] J. Yang and J. Leskovec, "Patterns of temporal variation in online media," in *Proc. Int. Conf. Web Search and Data Mining*, Feb. 2011, pp. 177–186.

[141] S.-W. Son, C. Christensen, G. Bizhani, D. V. Foster, P. Grassberger, and M. Paczuski, "Sampling properties of directed networks," *APS Physical Review E*, vol. 86, no. 4, pp. 046 104:1–12, Oct. 2012.

[142] N. Duffield, C. Lund, and M. Thorup, "Estimating flow distributions from sampled flow statistics," *IEEE Trans. Networking*, vol. 13, no. 5, pp. 933–946, Oct. 2005.

[143] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the *EM* algorithm," *J. Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[144] S. Borman, "The expectation maximization algorithm: A short tutorial," pp. 1–9, Jan. 2009. [Online]. Available: http://www.seanborman.com/publications/EM_algorithm.pdf

[145] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, 2nd ed. John Wiley & Sons, 2008.

[146] B. A. Huberman, D. M. Romero, and F. Wu, "Crowdsourcing, attention and productivity," *J. Information Science*, vol. 35, no. 6, pp. 758–765, Dec. 2009.

[147] S. Milojević, "Power-law distributions in information science — making the case for logarithmic binning," *J. American Society for Information Science and Technology*, vol. 61, no. 12, pp. 2417–2425, Dec. 2010.

[148] N. L. Johnson, A. W. Kemp, and S. Kotz, *Univariate Discrete Distributions*, 3rd ed. John Wiley & Sons, Inc., 2005, sec. 1.2.13.

[149] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *SIAM Review*, vol. 51, no. 4, pp. 661–703, 2009.

[150] W. E. Stein and R. Dattero, "A new discrete Weibull distribution," *IEEE Trans. Reliability*, vol. R-33, no. 2, pp. 196–197, June 1984.

[151] T. Nakagawa and S. Osaki, "The discrete Weibull distribution," *IEEE Trans. Reliability*, vol. R-24, no. 5, pp. 300–301, Dec. 1975.

[152] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1–2, pp. 81–93, June 1938.

[153] A. J. Lotka, "The frequency distribution of scientific productivity," *J. Washington Academy of Sciences*, vol. 16, no. 12, pp. 317–324, 1926.

[154] W. J. Reed and M. Jorgensen, "The double pareto-lognormal distribution—a new parametric model for size distributions," *Communications in Statistics - Theory and Methods*, vol. 33, no. 8, pp. 1733–1753, Apr. 2004.

[155] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 590–512, Oct. 1999.

[156] S. N. Dorogovtsev and J. F. F. Mendes, "Scaling behavior of developing and decaying networks," *Europhysics Ltrs.*, vol. 52, pp. 33–39, Oct. 2000.

[157] S. N. Dorogovtsev and J. F. F. Mendes, "Evolution of networks," *Advances in Physics*, vol. 51, no. 4, pp. 1079–1187, June 2002.

[158] L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, and L. Zhao, "Statistical analysis of a telephone call center," *J. American Statistical Association*, vol. 100, no. 469, pp. 36–50, 2005.

[159] A.-L. Barabási and J. G. Oliveira, "Human dynamics: Darwin and Einstein correspondence patterns," *Nature*, vol. 437, no. 7063, p. 1251, Oct. 2005.

[160] U. Harder and M. Paczuski, "Correlated dynamics in human printing behavior," *Physica A: Statistical Mechanics and its Applications*, vol. 361, no. 1, pp. 329–336, Feb. 2006.

[161] K.-I. Goh and A.-L. Barabási, "Burstiness and memory in complex systems," *Europhysics Ltrs.*, vol. 81, no. 4, p. 48002, Feb. 2008.

[162] D. Chakrabarti, Y. Zhan, and C. Faloutsos, "R-MAT: A recursive model for graph mining," in *Proc. Int. Conf. Data Mining*, Apr. 2004, pp. 442–446.

[163] B. Bollobás, C. Borgs, J. Chayes, and O. Riordan, "Directed scale-free graphs," in *Proc. Symp. Discrete Algorithms*, Jan. 2003, pp. 132–139.

[164] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on Twitter," in *Proc. Collaboration, Electronic Messaging, Anti-Abuse and Spam Conf.*, July 2010, pp. 1–9.

[165] J. Song, S. Lee, and J. Kim, "Spam filtering in Twitter using sender–receiver relationship," in *Proc. Int. Symp. Recent Advances in Intrusion Detection*, Sept. 2011, pp. 301–317.

[166] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu, "Analyzing spammer's social networks for fun and profit: A case study of cyber criminal ecosystem on Twitter," in *Proc. Int. World Wide Web Conf.*, Apr. 2012.

[167] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: Defending against Sybil attacks via social networks," *IEEE Trans. Networking*, vol. 16, no. 3, pp. 576–589, June 2008.

[168] M. P. H. Stumpf, C. Wiuf, and R. M. May, "Subnets of scale-free networks are not scale-free: Sampling properties of networks," *Proc. National Academy of Sciences of the United States of America*, vol. 102, no. 12, pp. 4221–4224, Mar. 2005.

[169] M. Cha, A. Mislove, and K. P. Gummadi, "A measurement-driven analysis of information propagation in the Flickr social network," in *Proc. Int. World Wide Web Conf.*, Apr. 2009, pp. 721–730.

[170] R. Kumar, J. Novak, and A. Tomkins, "Structure and evolution of online social networks," in *Proc. Int. Conf. Knowledge Discovery and Data Mining*, Aug. 2006, pp. 611–617.

[171] S. Milgram, "The small-world problem," *Psychology Today*, vol. 1, no. 1, pp. 61–67, May 1967.

[172] J. Travers and S. Milgram, "An experimental study of the small world problem," *Sociometry*, vol. 32, no. 4, pp. 425–443, Dec. 1969.

[173] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, June 1998.

[174] J. Leskovec and E. Horvitz, "Planetary-scale views on a large instant-messaging network," in *Proc. Int. World Wide Web Conf.*, Apr. 2008, pp. 915–924.

[175] C. R. Palmer, G. Siganos, M. Faloutsos, C. Faloutsos, and P. B. Gibbons, "The connectivity and fault-tolerance of the Internet topology," in *Proc. Wkshp. Network-Related Data Management*, May 2001, pp. 1–6.

[176] C. Bauckhage, K. Kersting, and B. Rastegarpanah, "The Weibull as a model of shortest path distributions in random networks," in *Proc. Wkshp. Mining and Learning with Graphs*, Aug. 2013, pp. 1–6.

[177] M. E. J. Newman, "Assortative mixing in networks," *Physical Review Ltrs.*, vol. 89, no. 20, pp. 208 701:1–4, Nov. 2002.

[178] J. G. Foster, D. V. Foster, P. Grassberger, and M. Paczuski, "Edge direction and the structure of networks," *Proc. National Academy of Sciences of the United States of America*, vol. 107, no. 24, pp. 10 815–10 820, June 2010.

[179] H.-B. Hu and X.-F. Wong, "Disassortative mixing in online social networks," *Europhysics Ltrs.*, vol. 86, no. 1, pp. 18 003:1–6, Apr. 2009.

[180] M. Kaiser, "Mean clustering coefficients: the role of isolated nodes and leafs on clustering measures for small-world networks," *New J. Physics*, vol. 10, no. 8, pp. 083 042:1–12, Aug. 2008.

[181] G. Fagiolo, "Clustering in complex directed networks," *APS Physical Review E*, vol. 76, pp. 026 107:1–8, Aug. 2007.

[182] K. Thomas, C. Grier, D. Song, and V. Paxson, "Suspended accounts in retrospect: An analysis of Twitter spam," in *Proc. Internet Measurement Conf.*, Nov. 2011, pp. 243–256.

[183] X. Chen, R. Chandramouli, and K. Subbalakshmi, "Scam detection in Twitter," in *Proc. Text Mining Wkshp.*, Apr. 2011, pp. 1–10.

[184] M. McCord and M. Chuah, "Spam detection on Twitter using traditional classifiers," in *Proc. Int. Conf. Automatic and Trusted Computing*, Sept. 2011, pp. 175–186.

[185] K. Thomas, C. Grier, and V. Paxson, "Adapting social spam infrastructure for political censorship," in *Proc. Wkshp. Large-Scale Exploits and Emergent Threats*, Apr. 2012.

[186] A. H. Wang, "Don't follow me: Spam detection in Twitter," in *Proc. Int. Conf. Security and Cryptography*, July 2010, pp. 1–10.

[187] C. Yang, R. C. Harkreader, and G. Gu, "Die free or live hard? empirical evaluation and new design for fighting evolving Twitter spammers," in *Proc. Int. Symp. Recent Advances in Intrusion Detection*, Sept. 2011, pp. 318–337.

[188] P. Resnick and R. Sami, "The influence limiter: Provably manipulation-resistant recommender systems," in *Proc. Conf. Recommender Systems*, Oct. 2007, pp. 25–32.

[189] P. Resnick and R. Sami, "The information cost of manipulation-resistance in recommender systems," in *Proc. Conf. Recommender Systems*, Oct. 2008, pp. 147–154.

[190] P. Resnick and R. Sami, "Sybilproof transitive trust protocols," in *Proc. Conf. Electronic Commerce*, July 2009, pp. 345–354.