Supporting Large Scale Communication Systems on Infrastructureless Networks Composed of Commodity Mobile Devices: Practicality, Scalability, and Security

by

Yue Liu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in The University of Michigan
2016

Doctoral Committee:

Professor Robert P. Dick, Chair
Professor Mingyan Liu
Assistant Professor Harsha V. Madhyastha
Associate Professor Vijay Subramanian
Professor Dan S. Wallach

ACKNOWLEDGEMENTS

I would like to thank my adviser, Professor Robert P. Dick, for his invaluable guidance, advice, and support through my PhD study. He conceived the original ideas of this thesis, and helped me refine research problems and overcome technical barriers. His way of problem-solving and attitude towards work have also influenced my practice in research.

I am grateful for the work of my collaborators. Special thanks to David Bild for our inspirational discussions. He improved a critical algorithm in the Mason test. He also participated in the design, implementation, and deployment of various software critical to this research. Aso thanks to David Adrian, Gulshan Singh, Nate Jones, Rongrong Tao, Jonathon Tiao, Anthony Tesija, and Junzhe Zhang for helping in the software development.

Many thanks to my co-advisors Prof. Dan S. Wallach and Prof. Z. Morley Mao. Their knowledge in security, distributed systems, and system design played important roles in shaping this research. Thanks to Prof. Mingyan Liu, Prof. Harsha V. Madhyastha, and Prof. Vijay Subramanian for serving on my committee. Your suggestions greatly improved several aspects of this work.

Special thanks to Walter Reynolds in the Information and Technology Services department of the University of Michigan. He provided the large-scale campus Wifi activity traces in Chapter IV. Without his generous help that research project would be impossible.

Finally, I must thank my family and friends for their continued and unconditional support and company. Without you it would be impossible for me to make this journey. I am lucky to have your love and support.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

Supporting Large Scale Communication Systems on Infrastructureless Networks Composed of Commodity Mobile Devices: Practicality, Scalability, and Security

by

Yue Liu

Chair: Robert P. Dick

Infrastructureless Delay Tolerant Networks (DTNs) composed of commodity mobile devices have the potential to support communication applications resistant to blocking and censorship, as well as certain types of surveillance. In this thesis we study the utility, practicality, robustness, and security of using these networks to provide useful and censorship-resistant communication services to local communities.

We collected two sets of wireless connectivity traces of commodity mobile devices with different granularity and scales for this study. The first dataset is collected through active installation of measurement software on volunteer users' personally owned smartphones. We implemented and deployed a prototype DTN microblogging application, called 1am, in the University of Michigan campus, and collected traces during a time period with 111 users. The second dataset is collected through passive observation of WiFi association events. It involves all 119,055 mobile devices recorded on the U-M campus WiFi network for a month.

Despite their differences in granularity and scales, simulation results show highly consistent message delivery performances of the two networks. Using an epidemic flooding protocol, the large network achieves an average delivery rate of 0.71 in 24 hours and 0.95 in 72 hours, with a median delivery delay of 10.9 hours. The small network achieves an average delivery rate of 0.7 in 24 hours and 0.8 in 72 hours, with a median delivery delay of 13 hours. We show that this performance is appropriate for sharing information that is not time sensitive, e.g., blogs and photos. Based on a measurement-based energy model, we also show that using an energy efficient variant of the epidemic flooding protocol, even the large network with 119,055 devices can support text messages while only consuming 13.7% of a typical smartphone battery in 14 hours.

We found that the network delivery rate and delay are robust to denial-of-service and censorship attacks. Attacks that randomly remove 90% of the network participants only reduce delivery rates to the remaining participants by less than 10%. Even when subjected to targeted attacks, the network suffered a less than 10% decrease in delivery rate when 40% of its participants were removed, and starts to break down only after 60% of the participants were removed.

In order to allow connectivity trace collection through active installation of measurement software, we developed a software platform to facilitate large-scale deployment-based studies of DTN protocols and applications on commodity mobile devices, named *MANES*. MANES provides an integrated, software-based framework that allows convenient control, observation, and logging of DTN communication and connectivity data during deployments. By providing emulated 802.11 ad hoc communications on commodity devices, MANES supports crowdsourcing deployments among common users without rooting their devices.

Although structurally robust, the openness of the proposed network introduces numerous security concerns since participants are not vetted. Ensuring security is further complicated because the system is distributed, without any central point of trust. The Sybil attack, in which a malicious node poses as many identities in order to gain disproportionate influence, is especially dangerous to distributed systems as it breaks the assumption underlying majority voting. Many defenses based on spatial variability of wireless channels exist, and we extend them to be practical for wireless ad hoc networks of commodity 802.11 devices without mutual trust. Specifically, we propose two efficient methods for separating valid channel measurement results of behaving nodes from those falsified by malicious participants. We present the Mason test, the first implementation of these techniques for ad hoc and delay-tolerant networks of commodity 802.11 devices.

CHAPTER I

Introduction

The Internet was once credited with promoting information sharing and access by connecting every corner of our planet. Yet, it can be used to facilitate surveillance and censorship at an unprecedented scale and granularity, undermining privacy and freedom of speech. With the knowledge of their customers, Internet giants like Google and Amazon monitor and record online activities for commercial purposes. Unauthorized mass information acquisition is also common. There is evidence that the NSA has continued engaging in "overcollection" of domestic communications since 9/11 [3, 4, 5].

In recent years, Internet censorship has been exercised with increasing frequency. During the Arab Spring, social media services such as Twitter and Facebook were intentionally blocked to quell protests [6]. In an extreme case, Internet service in Egypt was completely shut down for days in February 2011 [7]. These incidents are not singular. In 2012/2013 Reporters Without Borders has named 26 countries and 5 corporations conducting content filtering on the Internet via cyber attacks, or by pressuring technical service providers [8, 9].

We attribute the Internet's high vulnerability to surveillance and censorship to its hierarchical structure. Various research has shown that the connectivity graph of Internet routers is scale-free [10, 11]. The Internet becomes fragmented when less than 5% of its most connected routers are removed [12]. The same imbalance in traffic distribution is also observed in Internet applications. For example, in 2014, Google, Microsoft Bing, and Yahoo in total served 96.4% of all the search traffic in the U.S. [13] Because backbone routers and servers are usually owned by the government or a few large corporations, it is comparatively easy for powerful entities to gain access and control.

1.1 Censorship- and Surveillance-Resistant Techniques on the Internet

Security researchers and practitioners have designed many protocols and tools to protect privacy on the Internet, and to increase the cost of surveillance and censorship. Encryption is used across multiple layers of the TCP/IP network to protect the confidentiality and privacy of communications. *Internet Protocol Security* protects the confidentiality and authenticity of IP packets between network routers [14]. *Transport Layer Security* provides end-to-end encryption of traffic streams between application entities, e.g., between HTTP clients and servers [15]. In the application layer, tools like *GnuPG* [16], *Off-The-Record Messaging* [17], *Silent Text* [18], and *Cryptocat* [19] support confidential communication between end users.

Eventually, the security of encryption solutions depends on the security of their underlying key exchange/storage schemes, as whoever possesses the key gains access to the information. Any system that relies on a single third party for key distribution, e.g., a host server or a public key infrastructure, is vulnerable to attacks on that party. For example, Hushmail broke its promise to secure its clients' keys under the pressure of a court order [20]. DigiNotar, a trusted certificate authority in Netherlands, issued more than 500 false certificates in an attack in 2011 that caused more than 300,000 Iranian IP addresses to be compromised [21]. Decentralized key distribution schemes that rely on the web of trust, e.g., PGP [22], or manual verification between the two parties, e.g., voice-based verification, are more secure. Unfortunately, these systems are never popular because they are hard to use.

Anonymity techniques further protect communication metadata, i.e., the identities of involved parties. For example, *Tor* is a widely used tool to anonymize TCP streams via onion routing [23]. This technique is useful in combating censorship; seemingly benign traffic to uncensored proxies can be redirected to their real destinations, e.g., censored websites hosted outside the restricted region. Anonymous peer-to-peer file sharing techniques, e.g., Freenet [24] and RetroShare [25], are also useful in circumventing censorship. It is important to note that the degree of anonymity eventually depends on how large the suspect pool is. For example, the student who sent anonymous email bomb threat to Harvard University via Tor was quickly identified, not because of Tor's failure, but because the total number of Tor visitors in Harvard that day was probably very small [26].

An important lesson learned from reviewing these techniques is security solutions that put all trust on single parties are vulnerable and decentralized systems are more secure in face of attacks or coercion. With this observation, we push the effort of decentralization to an even lower level, the communication network itself. We move from the traditional Internet,

which is highly centralized, and explore the idea of using a fully decentralized network composed of people's own computing devices for secure, private, and censorship-resistant communication.

1.2 Infrastructureless Networks for Censorship and Surveillance Resistant Communication

With the commoditization of computers and wireless technology in recent years, researchers and practitioners have started to explore the idea of providing censorship- and surveillance-resistant communication through distributed, self-organizing wireless networks composed of participants' own devices [27, 28, 29, 30, 31]. The basic idea is to remove reliance on Internet infrastructure that is controlled by a very small group of companies and governments, and instead organize the computation and communication resources of the general public into an open, available, and robust communication network. It is difficult to monitor, access, or control traffic flows in this network as it requires controlling a large portion of the participating devices owned by the general public; it is expensive to do so.

One particular class of infrastructureless networks caught our attention, the so-called Pocket Switched Networks (PSNs), which are composed of people's own mobile devices, e.g., smartphones, tablets, and laptops [32, 33, 30]. Devices on these networks use their existing short range wireless modules, e.g., 802.11 ad hoc or BlueTooth, to exchange information upon contact. These networks have one distinct advantage: there is little economic or technical threshold to join. Anyone with a smartphone can participate by simply installing a software. As a result, their potential users are the whole pool of ever growing smartphone/laptop users who are eager to try out new technologies. In fact, during Hong Kong's Umbrella Revolution, the overnight success of Firechat, a smartphone app that uses PSNs to provide communication in the face of Internet failures [30], showed the interest in such networks. The application was downloaded 200,000 times between 28 and 30 September 2014 in response to the threat of potential denial of service attacks on the Internet-based Instagram [34].

For their practicality, we envision using PSNs to serve the communication needs of a local community such as a town or a university campus. The ultimate goal is to build an end-to-end solution to provide secure, private, and censorship-resistant communication services. As a first step towards this goal, a main focus of this thesis is enhancing our understanding of the performance and capacity of PSNs, especially when serving a whole community with hundreds of thousands of users. As PSNs are a class of Mobile Ad hoc and Delay Tolerant Networks, we will spend Section 2.3 to review related work. In Section 1.4

we go into more details on the specific problems with studying large-scale PSNs.

1.3 MANET and DTN Background

The concept of Mobile Ad hoc Networks (MANETs) emerged in the 1970s to describe infrastructureless wireless networks composed of mobile nodes, which are featured by continuous topology changes due to node mobility [35]. MANETs assume end-to-end connectivity between network nodes. On the other hand, Delay Tolerant Networks (DTNs), also referred to as Disruption Tolerant Networks, do not assume continuous network connectivity. The concept was first developed for deep-space communication, and later expanded to describe any infrastructureless networks with "limited expectations of end-to-end connectivity and node resources" [36]. The Pocket Switched Networks we work with, those composed of handheld devices equipped with commodity short-range radios, e.g., 802.11 ad hoc and Bluetooth, are best modeled by DTNs, because end-to-end connectivity is not guaranteed and human mobility has a large impact on the propagation of messages in these networks. We will use PSNs and DTNs interchangeably in the rest of this thesis.

Routing is the most fundamental problem of DTNs and previous research has made significant progress. Mayer classifies DTN routing protocols into three categories based on how forwarding decisions are made [37]. *Destination-aware* protocols use routing information, possibly containing indirect node encounters, to construct paths with highest delivery probability to the destination, e.g., PROPHET [38]. *Self-aware* protocols use only local information, e.g., each candidate's number of contacts, to make forwarding decisions independent of the destination, e.g., SimBet [39] and BUBBLE Rap [33]. *Unaware* protocols make no evaluation of candidate next-hops and simply forward messages to anyone in contact, e.g., epidemic flooding [40]. Our solutions build upon these existing routing protocols.

In the last decade with the emergence of high-performance mobile computing devices, e.g., handhold devices and laptops, the idea of connecting these devices into DTNs has gained increasing interest from the research community [32, 41, 42, 33, 43]. The main application scenario envisioned is to complement infrastructure wireless networks. Understanding human mobility and contact patterns is critical to understanding such networks. Towards this end, various deployments have been conducted and the collected human contact traces—either Bluetooth contacts [32, 44], or estimated contacts based on WiFi association records [45, 46], or course schedules [47]—shared with the research community. Researchers have found that human proximity networks closely resemble social networks with their small-world, community-based structures [48, 49, 50, 51, 33]. These new dis-

coveries have guided researchers to design more efficient routing protocols [39, 33] and applications [52].

1.4 Practicality and Scalability of PSNs

The application scenario we envision is a large-scale PSN that serves the communication needs of a local community, e.g., a small town or a university campus. Information, knowledge, and ideas that would otherwise be censored on the Internet are free to flow on this network. It is not the small scale ad hoc network for multi-player gaming or connecting to nearby printers that we commonly encounter. A communication system like this needs hundreds of thousands of users to survive and thrive. Therefore, it should be designed to support large numbers of users, e.g., a substantial percent of a town or a campus.

Due to the difficulty of collecting large-scale human mobility and contact traces, little is known about the performance and behavior of large-scale PSNs. In particular, we are interested in three basic questions concerning their practicality and scalability. First, what is the message delivery performance in terms of delivery rates and delay? Second, what is the overhead for each participant to support such a network? Third, how does this network react to changing adoption rate and random or intentional node failures?

Existing work mainly uses mathematical models to understand large-scale DTNs, most based on Markov chains [53, 41, 54] or fluid approximations [55, 56]. For analytical tractability, Poisson processes are used to model the contacts of node pairs and all pairs are assumed to follow independent and identical Poisson processes with the same contact rate [53]. Measurement results with real human contact traces contradict these assumptions. Inter-contact times are shown to follow power-law distributions [41, 57], contradicting the Poisson process assumption. Also, as shown in Chapter IV, different node pairs have very different contact rates.

We choose to study large-scale PSNs using real human mobility and contact traces. It is a non-trivial undertaking to collect large-scale human contact traces. There are various large-scale mobility traces generated from phone records [58] or smartphone application data [59], for example. However, their granularity is too coarse to infer direct contacts that are within 802.11 ad hoc or BlueTooth communication ranges. On the other hand, various human contact traces exist, but their scales are small, often a few hundred nodes [32, 44, 45, 46, 47]. The largest one we are aware of includes around 22,000 students on a university campus [47]. However, the contacts were derived from course schedules and therefore lack important details of the contact patterns. In addition, this dataset does not include contacts outside of classrooms, e.g., those in public libraries.

We used two methods to collect human contact traces. Both derived contacts based on WiFi association records. In the first case, we recruited volunteers to install study software on their smartphones. The number of participants was limited: 111. However, since the measurement software was installed on participants' personal smartphones, we were able to obtain a full history of their mobility and contacts (see Chapter IV). For the data collection we implemented a deployment and trace collection platform for MANET and DTN protocols, in hope of reducing overheads for deployment-based studies (see Chapter III).

In the second case, we passively monitored a university's campus WiFi network and obtaining all association/disassociation histories from the WiFi network controllers. This is the same method used in collecting the Dartmouth traces [45]. We recorded 119,055 mobile devices, nearly twice the campus population, indicating high coverage ratio of the population. These traces allows us to answer the original question on the performance and overhead of a very large-scale PSN (see Chapter IV). Note that this method is not able to record contact opportunities outside of the campus.

In addition to measuring and characterizing basic network performance, we also look into security problems on PSNs. We study how they react to random and targeted node failures using trace-driven simulations. We also look into one specific attack, the Sybil attack, which is particularly harmful in distributed systems. We design a practical Sybil defense that works with a completely distributed trust model (see Chapter II).

Our measurement studies on large-scale DTNs composed of commodity wireless devices show that these networks could deliver messages within a few hours to one or two days, and are therefore able to serve applications whose information is not time sensitive, e.g., photo sharing and weblogs. We also find that the energy overhead for participating in these networks is acceptable, e.g., supporting text messages consumes about 14% of a typical smartphone battery in 14 hours. On the other hand, we have shown that these networks are robust to random node failures and intentional censorship or blocking attacks; even in the worst attacks more than 40% of network nodes have to be eliminated to noticeably degrade the network performance. Our work in Chapter II further proves that it is possible to design complicated security protocols on these networks with a completely distributed trust model.

To summarize, we make the following statement according to findings of this thesis.

Thesis Statement: It is practical to use infrastructureless networks composed of commodity handheld devices to serve local communities (of hundreds of thousands of people) with secure and censorship-resistant communication applications that could tolerate comparatively longer delays, e.g., blogs and photo sharing.

1.5 Dissertation Outline

The rest of this thesis is organized as follows.

In Chapter II, we propose a practical solution to a common security problem in distributed wireless networks, the *Sybil attack*, in which a malicious node poses as many identities in order to gain disproportionate influence. Many defenses based on spatial variability of wireless channels exist, but depend either on detailed, multi-tap channel estimation—something not exposed on commodity 802.11 devices—or valid RSSI observations from multiple trusted sources, e.g., corporate access points—something not directly available in ad hoc and delay-tolerant networks with potentially malicious neighbors. We extend these techniques to be practical for wireless ad hoc networks of commodity 802.11 devices. Specifically, we propose two efficient methods for separating the valid RSSI observations of compliant nodes from those falsified by malicious participants. Further, we note that prior signalprint methods are easily defeated by mobile attackers and develop an appropriate challenge-response defense. Finally, we present the Mason test, the first implementation of these techniques for ad hoc and delay-tolerant networks of commodity 802.11 devices. We illustrate its performance in several real-world scenarios.

In Chapter III, we describe MANES, an emulation system for MANETs and DTNs composed of commodity mobile devices such as smartphones. Despite the increasing popularity of mobile computing devices, designing MANETs and/or DTNs applications and protocols that work under realistic conditions remains challenging because it is difficult and expensive to do large-scale deployment-based studies. MANES provides an integrated, software-based framework that allows convenient control, observation, and logging of communication data during deployments. By providing emulated 802.11 ad hoc communications on commodity devices, MANES supports crowdsourcing deployments with common users.

Chapter IV describes in detail the two sets of wireless connectivity traces of commodity mobile devices that we collected, including the collection methodology and basic statistics. We attempted two methods to collect fine granularity, large scale mobile device connectivity traces. The first method is active installation of measurement software on volunteer users' personally owned smartphones. We developed and deployed a prototype microblogging application on Android, named *1am* [60]. 1am had 291 users, with 111 users during a month of high activity on which our analysis focuses. We call this dataset "the 1am dataset". The second method is passive collection of WiFi association records, and infer direct 802.11 contacts using the following heuristics; two devices associated with the same access point simultaneously are considered in contact. We collected the contact traces of all 119,055 mobile devices recorded by the U-M campus WiFi network for a month. We call this dataset

"the U-M dataset".

In Chapter V we report measurement results of message delivery performance, energy overhead, and robustness to resource removal and censorship attacks for the two datasets. Despite their differences in granularity and scales, networks in the two datasets show similar performances. For example, the U-M network achieves an average delivery rate of 0.95 in 72 hours (0.71 in 24 hours) and a median delivery delay of 10.9 hours. The 1am network achieves an average delivery rate of 0.7 in 24 hours and 0.8 in 72 hours, with a median delivery delay of 13 hours. We show that using an energy efficient variant of the epidemic flooding protocol, the U-M network could support text messages with only consuming 13.7% of a typical smartphone battery in 14 hours.

Finally, we conclude the thesis with discussions on our discoveries and implications for future work in Chapter VI.

1.6 Contributions

We make the following contributions in this thesis.

- We design two methods of $O(n^3)$ complexity to separate true and false RSSI observations, enabling signalprint-based Sybil detection in ad hoc networks of nodes without mutual trust. The first method gives partial separation, bounding the number of misclassified identities. The second provides full separation, but works only when conforming nodes outnumber physical attacking nodes.

- We develop a challenge-response protocol to detect attackers attempting to use motion to defeat 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 and characterize its performance in real-world scenarios.

- We describe the design and implementation of MANES, a low-overhead, software based solution that facilitates experimental deployment to evaluate MANET and DTN protocols and applications using commodity mobile devices.

- We present a new technique to predict pairwise 802.11 connectivities using environmental Wi-Fi scan results. We show that this prediction technique yields significantly more accurate results than the widely used shared access point heuristic, according to an experimental evaluation with 17 users for more than 2,000 hours.

- We construct a connectivity graph of human location proximity involving 119,055 mobile devices on a university campus from their WiFi association histories for a month. We study basic properties of this human contact graph, reporting significantly higher node degrees (2863), shorter path lengths (2.3), and higher cluster coefficient (0.357) than popular online social networks. We show that its degree distribution has an exponential, instead of power-law tail. We also report that the human contact network shows strong "assortative mixing".

- Our analysis indicates that in a college town in which 0.2% of the population install a DTN app, the system has a median delivery rate of 0.85 after 72 hours and a median delivery delay of 13 hours. When all 119,055 mobile devices are involved, the campus-wide DTN achieves a message delivery rate of 0.71 after 24 hours, and 0.95 after 72 hours. The median delivery delay is 10.9 hours. We pay special attention to the system's performance variations and study their causes.

- We found that the delivery delays of DTN flooding protocols follow a power law distribution that varies from hours to days. We compare the distribution of the network's message delivery delays with the distributions of user response delays of various popular online applications, and show that this campus-wide DTN is appropriate for sharing non-time sensitive information such as Flickr photos and weblog articles.

- Robustness to denial-of-service and censorship attacks: We found that attacks that randomly remove 90% of the network participants only reduce delivery rates to the remaining participants by less than 10%. Targeted attacks, which remove nodes that have most contacts, were more harmful, causing the delivery rate to decrease precipitously when half of the participants were removed. However, even when subjected to targeted attacks, the network only suffered a less than 10% decrease in delivery rate when 40% of its participants were removed.

- We analyzed the impact of participating in large scale DTNs (119,055 nodes) on smartphone battery lifespan. Based on a measurement-based wireless communication aware smartphone power consumption model, our analysis shows that supporting text messages using a naive epidemic flooding protocol consumes 87.5% of a typical smartphone battery in 14 hours. We show that an energy efficient variant of the epidemic flooding protocol has around 10X decrease in energy consumption, while keeping comparable message delivery performance. Supporting text messages using this protocol consumes 13.7% of a typical smartphone battery in 14 hours.

CHAPTER II

The Mason Test: A Defense Against Sybil Attacks in Wireless Networks Without Trusted Authorities

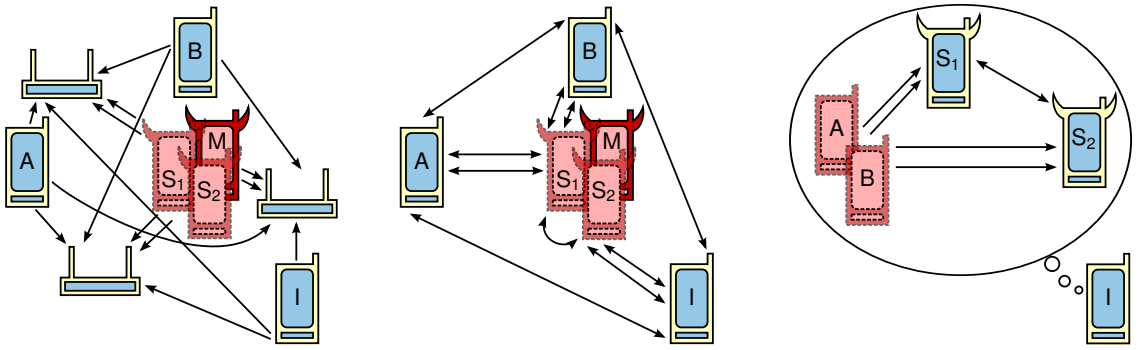
2.1 Introduction

The open nature of wireless ad hoc networks (including delay-tolerant networks [33]) enables applications ranging from collaborative environmental sensing [61] to emergency communication [62], 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 [63]. Newsome et al. survey Sybil attacks against various protocols [64], illustrating the need for a practical defense.

Proposed defenses (see Levine et al. for a survey [65]) fall into two categories. *Trusted certification* methods [66, 67] use a central authority to vet potential participants and thus are not useful in open ad hoc (and delay-tolerant) networks. *Resource testing* methods [68, 69, 70, 71] 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 greater than half the transmission wavelength [72] (6 cm for 2.4 GHz 802.11), so two transmissions with the same channel response are very likely to be from the same location and device [73, 1]. Note that two transmitters may be close enough, i.e., ~6 cm, to produce the same channel response, but this case is rare in practice.[1] One class of Sybil defenses

[1] In our experiments with smartphone users, distinct transmitters displayed similar channel responses in



(a) RSSI observations from trusted APs identify the Sybils, S_1 and S_2, presented by 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 2.1: Prior work [1, 2] assumes trusted RSSI observations, which are 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.

based on this observation uses specialized hardware to accurately measure and compare channel responses [1]. However commodity devices are not equipped with such hardware.

Commodity devices expose an aggregate, scalar value, the received signal strength. RSSI can be changed by varying transmit power, so a vector of observations from multiple receivers—a *signalprint*—is used instead, as its direction stays unchanged. Several authors have proposed such methods [2, 74, 75, 76] assuming trusted, true observations from, for example, access points (Figure 2.1a). In open ad hoc networks, observations are untrusted, coming from potentially lying neighbors (Figure 2.1b). In this case observations falsified by attackers can lead to incorrect conclusions (Figure 2.1c). Trust-less methods have been proposed, but have various limitations (e.g., devices must have uniform transmit power [77] or the method may be used only in outdoor environments with predictable propagation ranges [78]). Instead, a general method to separate true and false observations is needed.

We observe that, with high probability, attackers cannot produce false observations that make conforming identities look Sybil, due to the unpredictability of wireless channels. We exploit this weakness to bound the number of misclassified identities. In cases where conforming nodes outnumber physical attacking nodes (a major motivating factor for the Sybil attack), we develop a notion of consistency that enables fully accurate classification.

Signalprint-based detection is easily defeated by nodes that change locations to pro-

fewer than 0.01% of cases (see Figure 2.15 in Section 2.9).

duce multiple signalprints. Most past work ignores this problem, assuming that all nodes, including attackers, remain stationary. Although reasonable for conforming nodes, e.g., most human-carried smartphones are stationary over short time-spans, this is too strong an assumption for attackers. We remove this restriction on the attack model and defeat moving attacks by detecting and rejecting moving nodes. The rejection is temporary. Nodes can be tested again once stationary.

To detect moving attackers, Xiao et al. noted that successive transmissions from the same stationary node should have the same signalprint, while attackers cannot quickly (i.e., in milliseconds) switch between precise positions and therefore have inconsistent signalprints [1]. They did not further develop or evaluate a method making use of this observation. We develop a challenge–response protocol from this idea and study its performance on real deployments.

At a high level, we seek to allow a wireless network participant to occasionally determine which of its one-hop neighbors are non-Sybil. Verified non-Sybil participants, uniquely identified by their public keys, may safely participate in other protocols. In mobile networks, the process must be repeated occasionally (e.g., once per hour) as the neighbors change. Safety is more important than system performance, so nearly all Sybil identities must be detected. In most applications, it is acceptable for some non-Sybils to be rejected, e.g., any that were moving during the test.

We make the following primary contributions.

- We design two methods of $O(n^3)$ complexity to separate true and false RSSI observations, enabling signalprint-based Sybil detection in ad hoc networks of nodes without mutual trust. The first method gives partial separation, bounding the number of misclassified identities. The second provides full separation, but works only when conforming nodes outnumber physical attacking nodes.

- We prove conditions under which a participant can fully separate true and false observations.

- We develop a challenge-response protocol to detect attackers attempting to use motion to defeat 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 and characterize its performance in real-world scenarios.

2.2 Related Work

Many Sybil defense techniques are built on resource testing of wireless channels, because placing transmitters in many locations is much more difficult than acquiring additional computation or memory resources. Xiao et al. observe that in OFDM-based 802.11 channels, 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 [1].

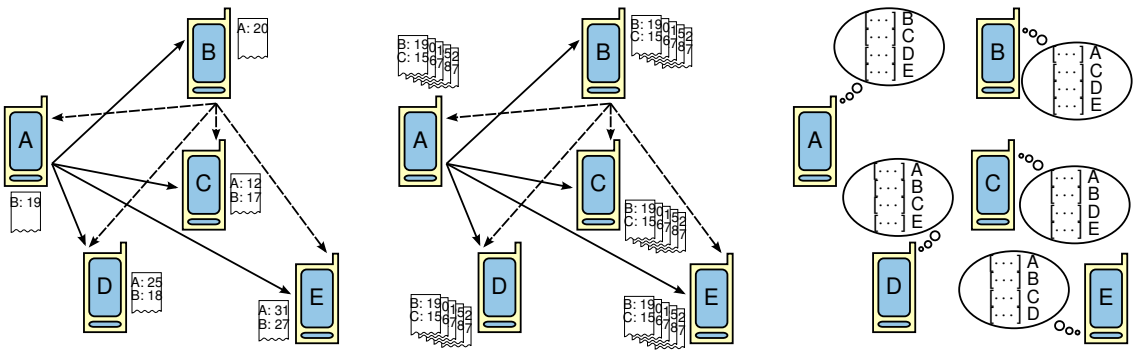
Li et al. use the unique mapping between identity and wireless channel to develop a channel-based authentication scheme, using both pulse-type probing in the time domain and multi-tone probing in the frequency domain for channel estimation [79]. Although not originally designed for Sybil defense, applying this technique to detect multiple identities sharing the same channel is straightforward. A primary drawback of this class of work is its restriction to specialized hardware or firmware, as commodity 802.11 devices do not expose detailed channel information to the driver and operating system.

Faria et al. and Demirbas et al. independently developed the signalprint technique, which greatly simplifies channel estimations while maintaining high Sybil detection performance [2, 74]. Instead of measuring probe responses, a vector of RSSIs reported by multiple receivers at different locations is used to characterize the sender's unique location and wireless environment.

This class of work [2, 74, 75, 76] has two disadvantages. First it relies on trusted external measurements, e.g., RSSIs from trusted 802.11 access points, which are generally unavailable in open ad hoc networks. Our work builds on their ideas, but does not rely on any particular external device being trustworthy. Second, it restricts the attack model to stationary devices, even though attackers can easily use mobile devices. Our work detects and rejects moving nodes, instead of accepting them as non-Sybil.

Lv et al. developed a method based on one-dimensional signalprints, which therefore does not rely on any external measurements [77]. However, it assumes, unrealistically, a uniform transmit power for all devices, including attacking devices.

Bouassida et al. developed a trust-less method for vehicular area networks. Instead of relying on external measurements, the verifier obtains uncorrelated measurements by changing its own reception locations. These measurements are used to locate the transmitter and detect abnormalities. It also rejects moving nodes with significant location changes over multiple measurements [78]. However, this technique relies on a predictable propagation model for location estimation that fails to capture the notorious variations of wireless channels. Our method does not assume any propagation model. Instead, we rely on the



(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 may lie about their observations.

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

Figure 2.2: The solution framework for signalprint-based Sybil detection in ad hoc networks. In this work we flesh out this concept into a safe and secure protocol, the Mason test.

unpredictability of wireless signal propagation to defeat lying attackers.

2.3 Problem Formulation and Background

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

2.3.1 Problem Formulation

Our goal is to extend signalprint-based Sybil detection methods to work without a priori trust in any observer, allowing any participant in an open wireless network to determine which of its one-hop neighbors are non-Sybil. The solution framework is illustrated in Figure 2.2 with five participants. We assume an arbitrary identity (or condition) starts the process. Participants first take turns broadcasting probe packets while all others record the observed RSSIs (Figure 2.2a). These observations are then shared, although malicious nodes may lie. Figure 2.2b shows every participant after this exchange, with observations from all five participants. Finally each participant individually selects a (hopefully truthful) subset of observers for signalprint-based Sybil classification (Figure 2.2c).

In this work we present our method for truthful subset selection and flesh 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 2.10 discusses these choices and related potential concerns.

2.3.2 Attack Model

We model attackers who operate commodity devices, but not specialized hardware. Commodity devices can be obtained, at a large scale, by compromising those owned by normal network participants, a more practical attack vector than distributing specialized hardware at the same scale. Specifically, 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 the RSSI observations of other nodes, e.g., 7 dBm uncertainty (see Section 2.6), 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, i.e., they are not equipped for antenna array-based beam-forming.
5. Attackers can move their devices, but cannot quickly and precisely switch them between multiple positions, e.g., they do not have high-speed, automated electromechanical control.

One common denial-of-service (DOS) attack in wireless networks, i.e., jamming the channel, cannot be defended against by commodity devices. Thus, we do not defend against other more-complicated DOS attacks. However, note that ad hoc and delay-tolerant networks are much more resistant than infrastructure networks to such attacks, because a single attack can affect only a small portion of the network. Moreover, DOS attacks are less catastrophic to privacy and security than successful Sybil attacks.

Notably, we assume attackers do not have per-antenna control of MIMO (Multiple-Input and Multiple-Output) [80] devices. Such control would defeat the signalprint method (even with trusted observers), but is costly to implement. 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 Appendix A for a proposed solution and its theoretical proof), but doing so is beyond the scope of this work. Our focus is extending signalprint-based methods to ad hoc networks of commodity devices by removing the requirement for trusted observations.

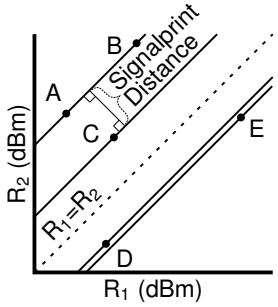


Figure 2.3: Sybils, *A*–*B* and *D*–*E*, occupy nearby slope-1 lines.

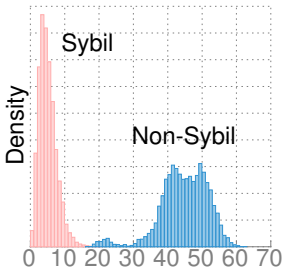


Figure 2.4: The classification threshold trades false positives for negatives.

2.3.3 Review of Signalprints

We briefly review the signalprint method. See prior work for details [1, 74]. 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 [72] 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 $\vec{\mathbf{s}}_\mathbf{b} = \vec{\mathbf{s}}_\mathbf{a} + c\vec{\mathbf{1}}$. In other words, all observers see the same RSSI difference c for the two transmissions.

This is illustrated geometrically in Figure 2.3 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 on 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

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

Figure 2.4 shows the distance distributions for Sybil and non-Sybil identities using measurement data for commodity Android devices.[2] The two distributions are well separated with small overlap, so the choice of classification threshold trades false positives for false negatives. A good threshold supports detection of at least 99.9% of Sybils while accepting at least 95% of non-Sybils, as reported by previous research [1, 74] and confirmed by our own measurement (see Figure 2.13).

2.4 Sybil Classification From Untrusted Signalprints

In this section we describe two methods to detect Sybil identities using untrusted RSSI observations. In both cases, a set of candidate views containing the true view (with high probability) is generated. The accepted view is chosen by a view selection policy. The first method selects the view indicting the most Sybils, limiting the total number of incorrect classifications. The second selects the true view, but works only when conforming nodes outnumber physical attacker nodes.

2.4.1 The Limited Power of Falsified Observations

Our key observation is that falsified RSSI observations have limited power. Although falsifying observations to make Sybil identities look non-Sybil is easy, it is extremely difficult to make a non-Sybil look Sybil. To see this, recall that a pair of identities is considered Sybil only if all observers, including the initiator itself, report the same RSSI difference for the pair's transmissions. Making true Sybils appear non-Sybil is easy, because randomly chosen values almost certainly fail to match the difference observed by the initiator. Making a non-Sybil look Sybil, however, requires learning the difference observed by the initiator, which is kept secret. Guessing is difficult due to the unpredictability of the wireless channels. Our methods rely on this difficulty. They are developed formally in the rest of this section. Quantitative characterizations are described in Section 2.6. To summarize, the success probability for a guessing attacker is less than 10^{-6} in common situations, e.g., when conforming nodes outnumber physical attackers by more than 1.53× (see Figure 2.8).

[2] We used size-4 signalprints from the "Outdoor" experiment in Section 2.9.

2.4.2 Terminology

Table 2.1 lists all the terms and symbols used in the development of the classification methods. 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 first knowing the L and T partitions.

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

Definition. A *receiver set*, denoted by R, is a subset of identities ($R \subseteq I$) whose reported RSSI observations, combined 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.

[3] A reported RSSI observation is considered *false* if some signalprints containing it misclassify some identities.

[4] All participants perform classification individually, so each is the initiator in its own classification session.

Table 2.1: 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	

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 Section 2.4.7, we extend the method to handle real-world observations containing, for example, random noise and discretization error.

2.4.3 Approach Overview

A general separation method does not exist, because different scenarios can lead to the same reported RSSI observations. To illustrate, consider identities $I = \{A|B\}$ reporting observations such that

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

and two different scenarios x and y such that

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

$R \subseteq T \mapsto \overline{V}$, so V^1 and V^2 are both true views, the former in scenario x and the latter in scenario y. In other words, identities in A could be Sybil (as claimed by B) or those in B could be Sybil (as claimed by A). Either view could be correct; it depends on which group is lying. Consequently, no method can always choose the correct view.

We instead develop two different approaches. The first method, the *maximum Sybil policy*, simply bounds the number of misclassified identities by selecting the view reporting the most Sybils. This selected view must indict at least as many as the true view, bounding the accepted Sybils by the number of collapsed conforming identities. Collapsing is difficult, limiting the number of incorrect classifications.

The second method, the *view consistency policy*, allows complete separation, but requires that the following conditions be met.

- All views correctly classify some conforming identities (likely true because collapsing identities is difficult).
- Conforming identities outnumber lying, non-Sybils (a major motivating factor for the Sybil attack).

This approach follows from the idea that true observations are trivially self-consistent, while lies often contradict themselves. We develop a notion of consistency that allows separation of true and false observations.

2.4.4 Maximum Sybil Policy: Select the View Claiming the Most Sybil Identities

In this section, we prove that the maximum Sybil policy—selecting the view claiming the most Sybil identities—produces a classification with bounded error. The number of incorrectly-accepted Sybil identities is bounded by the number of collapsed conforming identities.

Lemma 1. *The selected view V claims at least as many Sybil identities as actually exist, i.e.,* $|V_S| \geq |S|$.

Proof. Since the true view $\overline{V}$ claiming $|S|$ Sybils always exists, the selected view can claim no fewer. □

Theorem II.1. *The selected view V misclassifies no more Sybil identities than it collapses conforming identities, i.e.,* $|V_{NS} \cap S| \leq |V_S \cap NS|$.

Proof. Claiming the minimum $|S|$ Sybil identities requires that each misclassified Sybil be compensated for by a collapsed non-Sybil identity. Formally, combining $|V_S \cup V_{NS}| = |S \cup NS|$ with Lemma 1 yields $|(V_S \cup V_{NS}) \cap S| \leq |(S \cup NS) \cap V_S|$. Removing the common $V_S \cap S$ from both sides gives $|V_{NS} \cap S| \leq |V_S \cap NS|$. □

Theorem II.1 bounds the misclassifications by the attacker's collapsing power, $|V_S \cap NS|$. Although $|V_S \cap NS|$ is small (see Section 2.6), one Sybil is still accepted for each conforming identity collapsed. The next few sections develop a second method that allows accurate classification, but only when conforming nodes outnumber attackers.

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

Our view consistency policy stems from the intuition that lies told by those with incomplete information often contradict each other. It is introduced here using the following unrealistic assumption, which we remove in Section 2.4.6.

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 notion of 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 2. *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. However, a fully omniscient attacker could theoretically generate a false, consistent view by collapsing all conforming identities. In practice, the difficulty of collapsing identities prevents this. We formalize this attacker limitation 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 2.6 quantifies the probability that this condition holds.

Lemma 3. *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 II.2. *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 2 and Lemma 3, 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. Restriction 1 does not hold in practice, so we develop methods to relax it.

2.4.6 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 from consideration. 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 over 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 protocols require a majority of the nodes to conform. 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 2.6 quantifies the probability that it holds.

Lemma 4. *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 2. 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 II.3. *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 4 and Theorem II.2. □

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

2.4.7 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 [1, 74]), 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 $|C| > 20$) [1, 74].

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

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 $|V^1_{\text{NS}} \cap V^2_{\text{NS}}| \geq (1-2\gamma_n)|NS|$. Each misclassifies at most γ_n of the non-Sybil identities, so $|V^1_{\text{NS}} \setminus V^2_{\text{NS}}| \leq \gamma_n|NS|$ and similar for $V^2_{\text{NS}} \setminus V^1_{\text{NS}}$. 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 2.6. An analogue of Theorem II.3 follows.

Theorem II.4. *Under Condition 3, the γ_n-consistent view of the largest subset of I permitting such a view is γ_n-true.*

In Section 2.5 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] γ_n is an upper bound on the total fraction misclassified, not the probability that an individual identity is misclassified.

[6] This assumes that the false negative bound is negligible. If it is not, a similar notion of γ,σ-consistency, where σ is the false negative bound, can be used. In practice σ is quite small [1, 74], so simple γ_n-consistency is fine.

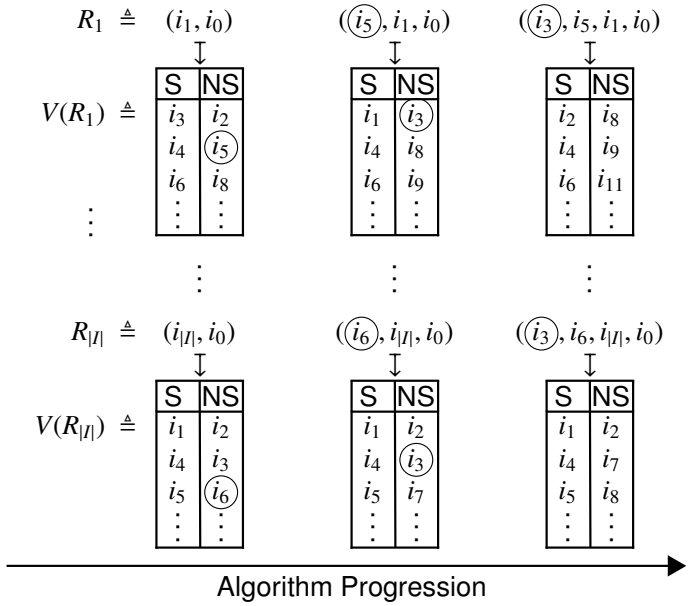


Figure 2.5: Illustration of Algorithm 1. All $|I|$ size-2 receiver sets are increased to size-4 by iteratively adding a random identity from those labeled non-Sybil by the current set. With high probability, at least one of the final sets will contain only conforming identities.

2.5 Efficient Implementation of the Selection Policies

Both the maximum Sybil and view consistency policies offer ways to select a view, either the one claiming the most Sybils or the largest one that is γ_n-true, but brute-force examination of all $2^{|I|}$ receiver sets is infeasible. Instead, we describe $O(|I|^3)$ algorithms for both policies. In summary, both start by generating $O(|I|)$ candidate views (Algorithm 1). For the maximum Sybil policy, the one claiming the most Sybil identities is trivially identified. For the view consistency policy, Algorithm 2 is used to identify largest γ_n-consistent view.

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: $\quad\quad R \leftarrow R \cup \{\text{RandElement}(V_{\text{NS}}(R))\}$
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

2.5.1 Candidate Receiver Set Selection

The only requirement for candidate receiver set selection is that at least one of the candidates 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 2.5, 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:

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

Figure 2.6 shows this probability as a function of the number of conforming identities ($|C|$) and the number of non-Sybil liars ($|LNS|$). We use size-4 signalprints ($n = 4$) and $\gamma_4 = 0.05$, based on previous evaluation results [1, 74]. In the shaded areas, some required condition is not met. Recall that Algorithm 1 requires $|C| > n$, so that at least one size-n receiver set composed purely of conforming nodes can be formed. The view consistency policy requires $|C| > |LNS|$ (Condition 2).

The signalprint threshold for this process is chosen to eliminate (nearly) all false negatives, because the goal is to minimize the malicious-to-conforming ratio; false positives are harmless during the generation of candidate views. The complexity of a straightforward implementation is $O(|I|^3)$. Section 2.10 further discusses the runtime.

2.5.2 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 II.4, is that permitting the largest subset of I to be γ_n-consistent. Checking consistency by examining all $2^{|V_{NS}|}$ 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). The view excluding the fewest is γ_n-true and the desired classification (line 22).

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

[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.

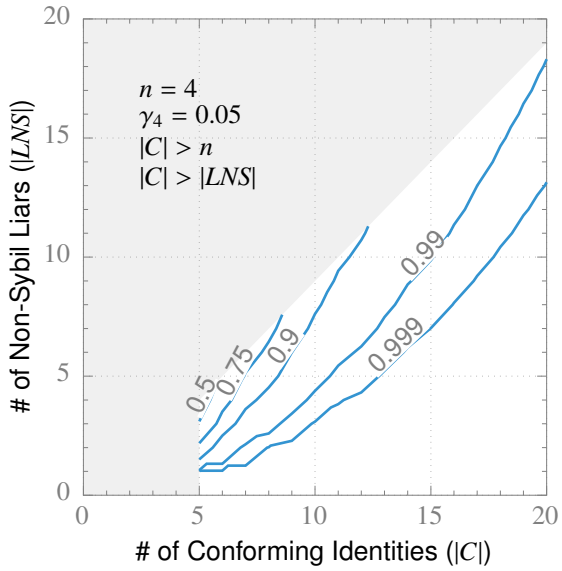


Figure 2.6: Contours of probability that at least one of the receiver sets from Algorithm 1 is conforming.[7] In the shaded areas, conditions required by either the consistency policy or by Algorithm 1 are not met.

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

$(C, R_{\max}) \leftarrow (\infty, \text{null})$
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**
    $(C, R_{\max}) \leftarrow (c, R)$ ▷ new largest γ-consistent subset found
  end if
end for
return $R_{\max}$

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 $V_S(R)$, the RSSI ratio with an arbitrary sibling (line 3) and comparing against those reported by potential non-Sybils in $V_{NS}(R)$ (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). It is also excluded if the receiver set consisting of just itself and the initiator is not γ_2-similar to R (line 11).

2.5.3 Runtime in the Absence of Liars

In a typical situation with no liars, the consistency 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.

2.6 Classification Performance Against Optimal Attackers

Both view selection policies depend directly on the unpredictability of RSSIs, because collapsing identities requires knowing the observations of the initiator, as explained in Section 2.4.1. An intelligent attacker can attempt educated guesses, resulting in some successful collapses. In this section, we evaluate the two selection policies against the optimal attackers, as defined in Sections 2.6.2 and 2.6.3.

2.6.1 RSSI Unpredictability

Accurately guessing RSSIs is difficult because the wireless channel varies significantly with small displacements in location and orientation (*spatial variation*) and environmental changes over time (*temporal variation*) [72, 81]. Pre-characterization could account for spatial variation, but would be prohibitively expensive at the needed spatial and orientation granularity (6 cm [82] and 3° for our test devices).

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

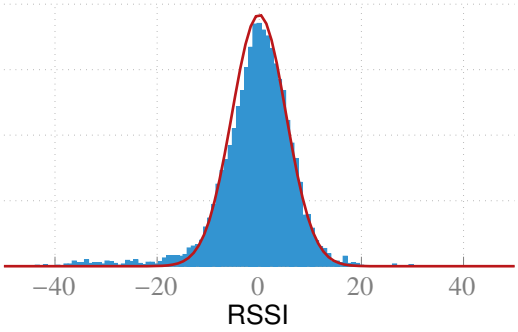


Figure 2.7: Distribution of RSSI variations in real-world deployment.

We empirically determined the RSSI variation for human-carried smartphones by deploying experimental phones to eleven graduate students in two adjacent offices and measuring pairwise RSSIs for fifteen hours. The observed distribution of deviations,[9] shown in Figure 2.7, 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 [72]). We use this distribution to model the attacker uncertainty of RSSIs, corresponding to an attacker who accumulates knowledge of pairwise RSSIs by observing values reported in past tests.

2.6.2 Optimal Attacker Strategy—Maximum Sybil Policy

Theorem II.1 shows that the performance of the maximum Sybil policy is inversely related to the number of collapsed non-Sybil identities. Therefore, the optimal attacker tries to collapse as many as possible. We give two observations about this goal.

1. More distinct guesses increase the probability of success, so an optimal attacker partitions its (mostly Sybil) identities, with each group making a different guess.

2. Smaller group size increases the number of groups, but decreases the probability that the group is considered—recall that Algorithm 1 generates only $|I|$ of the possible $2^{|I|}$ candidate receiver sets.

Consequently, there is an optimal group size that maximizes the total number of groups (guesses) produced by Algorithm 1, which we obtained via Monte Carlo simulations. We model the initiator's RSSI observation as a random vector whose elements are drawn i.i.d. from the Gaussian distribution in Figure 2.7. Given the total number of guesses, the best

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

choices are the vectors with the highest joint probabilities. The performance against this strategy is discussed in Section 2.6.4.

2.6.3 Optimal Attacker Strategy—View Consistency Policy

The view consistency policy depends on Condition 3 holding, i.e., all consistent views must correctly classify at least $|LNS| + 1$ conforming identities. In this section we quantify the probability that it holds against an optimal attacker. 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 values of the sets that are compatible. For example, the second most likely values for the set (a, b) are (-78 dBm, -49 dBm), and the second most likely values for the set (a, c) are (-82 dBm, -54 dBm). These two sets of values are incompatible, as one cannot simultaneously guess both -78 dBm and -82 dBm for node a.

The above problem is non-trivial, but an attacker could conceivably solve it. In order to model the strongest possible attack, we assume that all sets of values are compatible.

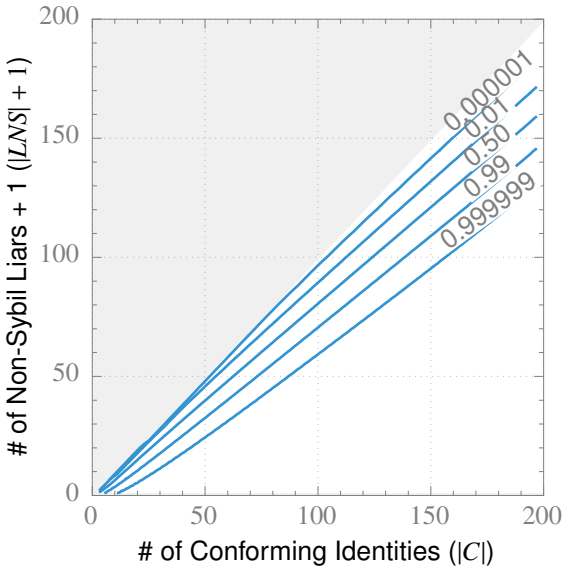


Figure 2.8: Contours of a lower bound on the probability that Condition 3 holds under an optimal attacker strategy with the attacker's knowledge of RSSIs modeled as a normal distribution with standard deviation 7.3 dBm.

For example, we assume one group can simultaneously guess (−78 dBm, −49 dBm) for the set (a, b), and (−82 dBm, −54 dBm) for the set (a, c). Any realizable attack would use an additional group to try both guesses. Thus, this assumption models an attack that, with the same set of groups, has a higher success probability than any realizable attack. This leads to a conservative lower bound on the probability that the attacker fails—any feasible, optimal strategy is less likely to succeed.

Figure 2.8 shows contours of this lower bound on the probability that Condition 3 holds as a function of $|C|$ and $|LNS|$, obtained via Monte Carlo simulations of the super-optimal attacker. The initiator's RSSI observation is modeled as a random vector, whose elements are drawn i.i.d. from the Gaussian distribution in Figure 2.7. The $|C| \leq |LNS|$ region is shaded, because the view consistency policy fails there (recall Condition 2). 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.

2.6.4 Performance Comparison of Both Policies

We use Monte Carlo simulations to compare the performance of the two policies against the optimal attackers, quantified as the *final Sybil ratio*, the fraction of accepted identities that are Sybil. We model the attacker's knowledge of the initiator's RSSIs as a random

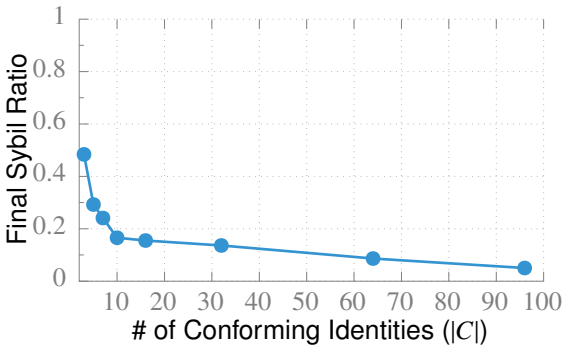


Figure 2.9: The final Sybil ratio, i.e., fraction of accepted identities that are Sybil, produced by the maximum Sybil policy against an optimal attacker strategy.

vector whose elements are drawn i.i.d. from the Gaussian distribution in Figure 2.7, which conservatively assumes fine-grained temporal and spatial characterization (see Section 2.6.1). We expect real-world attackers to have less knowledge, leading to even better classification performance.

Our procedure for generating candidate receiver sets (Algorithm 1) works best when conforming nodes outnumber physical attackers. This condition should normally hold in real-world networks (it is the major motivation for a Sybil attack), so for both policies, we report results assuming that it does.

Figure 2.9 graphs the final Sybil ratio of the maximum Sybil policy, which roughly corresponds to the ratio of collapsed conforming nodes ($\frac{|V_S \cap NS|}{|C|}$). The performance does not depend on the number of physical attackers. The Sybil ratio decreases to 0.05-0.2 when $|C| > 10$. When $|C| < 10$, the Sybil ratio is high (0.2–0.5), despite elimination of most Sybil identities (92%–99%). This behavior is due to the ease of guessing low-dimension random vectors.

Figure 2.10 shows the final Sybil ratio of the consistency policy. Again, the $|C| \leq |LNS|$ region is shaded as the policy simply fails in this case. Performance increases rapidly with the ratio of conforming nodes to physical attackers—recall the attacker needs to collapse $|C| - |LNS|$ identities to break Condition 3. For example, the final Sybil ratio drops below 10^{-6} when $\frac{|C|}{|LNS|+1} \geq 1.6$. As the collapse rate is usually below 0.2 (see Figure 2.9 when $|C| > 10$), we observe good performance when $|C| - |LNS| \geq 0.2|C|$ (below the 0.05 contour). The dashed line (roughly $\frac{|C|}{|LNS|+1} = 1.2$) indicates the situations where both policies perform equally. Below it, the consistency policy performs better than the maximum Sybil policy and above it does worse.

The view consistency policy is superior when conforming nodes are expected to outnumber attacker nodes by at least 1.2×, the common case in urban environments. The maximum

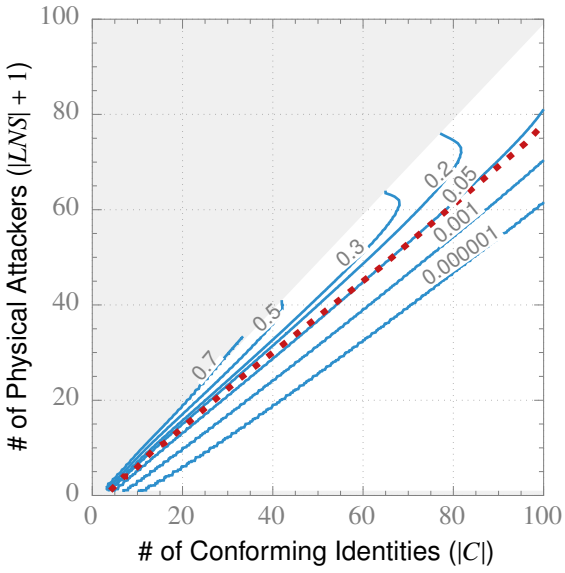


Figure 2.10: Contours showing the final Sybil ratio for the view consistency policy against an optimal attacker strategy. The dashed line corresponds to situations where this policy has the same performance as the maximum Sybil policy.

Sybil policy remains viable when the number of physical attackers is comparable to (or even larger than) that of the conforming nodes. We suggest users of the Mason test consider their application knowledge when choosing a policy.

2.7 Detecting Moving Attackers

A mobile attacker can defeat signalprint comparison by changing locations or orientations between transmissions to associate distinct signalprints with each Sybil identity. Instead of restricting the attack model to only stationary devices, we protect against moving attacks by detecting moving nodes. Moving nodes are treated as non-conforming, in essence, and will not be able to participate in network protocols until stationary enough to be tested for Sybilness again. 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. Thus, multiple transmissions should have the same signalprints [1]. 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 2.9, this simple criterion yields acceptable detection.

The protocol collection phase (Figure 2.2a) is extended to request multiple probe packets

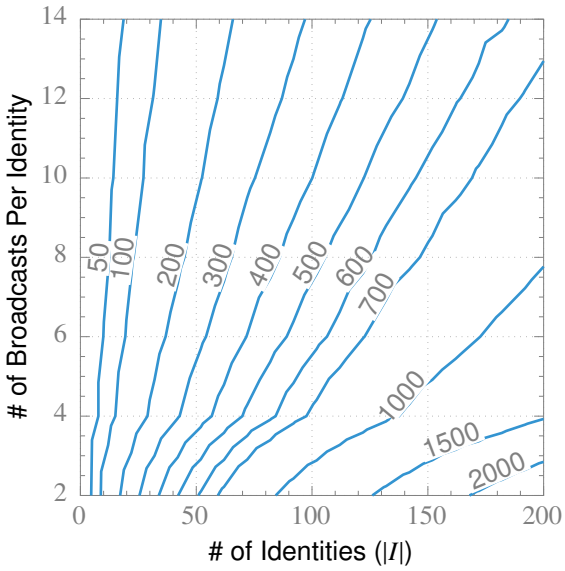


Figure 2.11: Contours showing the response time (in ms, 99$^{\text{th}}$ percentile) to precisely switch between two positions required to defeat the challenge-response moving node detection protocol.

(e.g., 14) from each identity in a pseudo-random order (see Section 2.8.1). During the classification phase (Figure 2.2c) each participant rejects any identity with a 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 cm location precision according to coherence length theory [82] and 3° orientation precision according to our measurements).

Figure 2.11 plots the required response time for an attacker to pass the challenge. Random sequences of probe requests are generated via Monte Carlo simulations and the required response time is calculated accordingly. Given human reaction times [83], 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.

2.8 The Mason Test

This section describes the full Mason test protocol, an implementation of the concepts introduced in the previous sections. There are 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 pseudo-random order. Further, each participant must be able to verify that no group of identities controlled the order.

3. Moving identities must be rejected. 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 constructing 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 2.10 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 protocol and leads the collection, but all identities still individually and safely perform Sybil classification.

2.8.1 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, each 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), the protocol aborts because an attacker may be selectively jamming some HELLO-Is. The protocol also aborts if too many identities join, e.g., 400.

Phase II: Randomized Broadcast Request: The second phase executes the challenge-response protocol to collect RSSI observations for motion detection and Sybil classification. First, each identity contributes a (difficult to predict) random value;[10] 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 each participant in the random sequence,

[10]Even if attackers do not comply, conforming participants can verify that their own random submissions resulted in a random sequence and therefore trust the test results.

who must quickly broadcast a signed HELLO-II, e.g., within 10 ms in our implementation.[11] 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 physical position and is rejected.

In some applications, it might be desirable to meet the additional requirement that attackers be unaware of their positions in the challenge-response sequence until challenged. This could be achieved by allowing the initiator to use a self-generated random sequence that cannot be verified by other participants. However, if this were done only the initiator would be able to safely use the test results.

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.

2.8.2 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 2.12). Thus, a node only performs Sybil classification if it has strong evidence that the current observations are uncorrelated with prior ones,[12] i.e., it has observed an acceleration of at least 2 m/s^2.

Classification is a simple application of the methods of Section 2.7 and Section 2.5. 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.

[11] 10 ms is larger than the typical roundtrip time for 802.11b with packets handled in interrupt context for low-latency responses. These packets can be signed ahead of time and cached—signatures do not need to be computed in the 10 ms interval.

[12] 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.

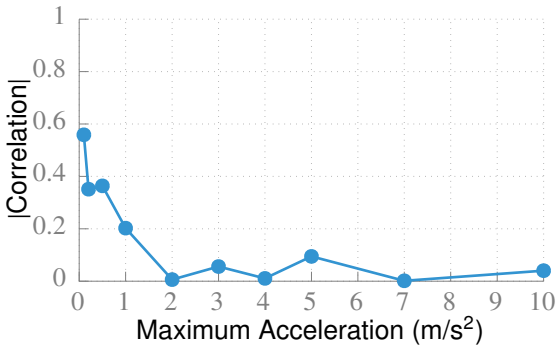


Figure 2.12: RSSI correlation as a function of the maximum device acceleration between observations.

2.9 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). The classification algorithms are implemented in Python. Unlike the described protocol, mobile conforming nodes participated in all tests (i.e., nodes did not monitor their own motion and decline to participate when moving), giving us data to tune the motion filter and characterize the impact of node motion on the classifier performance.

The goal of this section is to evaluate the overall performance of our system in normal settings, which is mainly dependent on the wireless environment. We therefore 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[13] generating 4, 20, 40, and 160 Sybil identities. The ratio of conforming to attacking nodes is held constant, as

[13] As discussed in Section 2.4 and Section 2.6, additional physical nodes are best used as lying, non-Sybils.

Table 2.2: 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

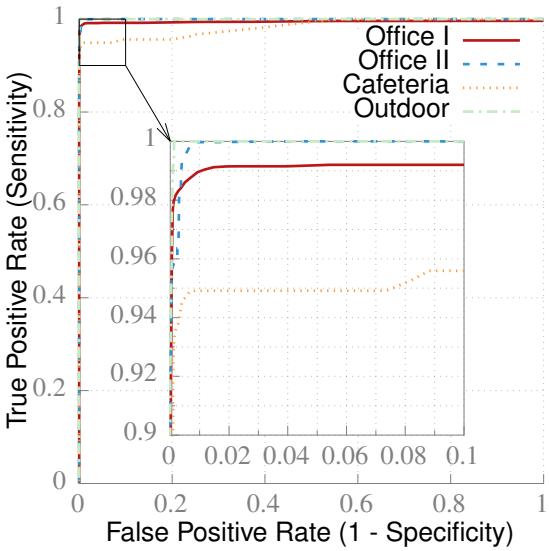


Figure 2.13: ROC curve showing the classification performance of signalprint comparison in different environments for varying distance thresholds. Only identities that passed the motion filter are considered. The knees of the curves all correspond to the same thresholds, suggesting that the same value can be used in all locations.

it does not affect performance (assuming at least one true view is generated by Algorithm 1). The gathered traces were split into testing and training sets.

We do not study the system performance under collapsing attacks here, as it also depends on the number of conforming and attacking nodes, and we have too few experimental devices to meaningfully vary those counts. In Section 2.6 we independently evaluate the performance against such attacks, using Monte Carlo simulations to vary both numbers from 5 to 200.

2.9.1 Selection and Robustness of Thresholds

The training data were used to determine good motion filter and signalprint distance thresholds, shown in Table 2.2.

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 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 2.13 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 [1, 74], as expected.

In all environments, the knees of the curves 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 work use these same thresholds, shown in Table 2.2.

2.9.2 Classification Performance

The performance of the full Mason test, including motion filtering and signalprint comparison, is detailed by the confusion matrices in Figure 2.14. Note that we count all rejected identities, including both Sybil and moving identities, as Sybil. Many tests were conducted in each 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 number of identities is limited (e.g., to 400), so rejecting most of the Sybils and accepting most of the conforming identities is sufficient to meet this requirement.

Table 2.3 shows the performance for all four environments. The Mason 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 motion of cafeteria patrons. In the outdoor environment, with all participants (including attackers) moving, the sensitivity is 95.9%, and the specificity is 61.1% with all the false rejections caused by

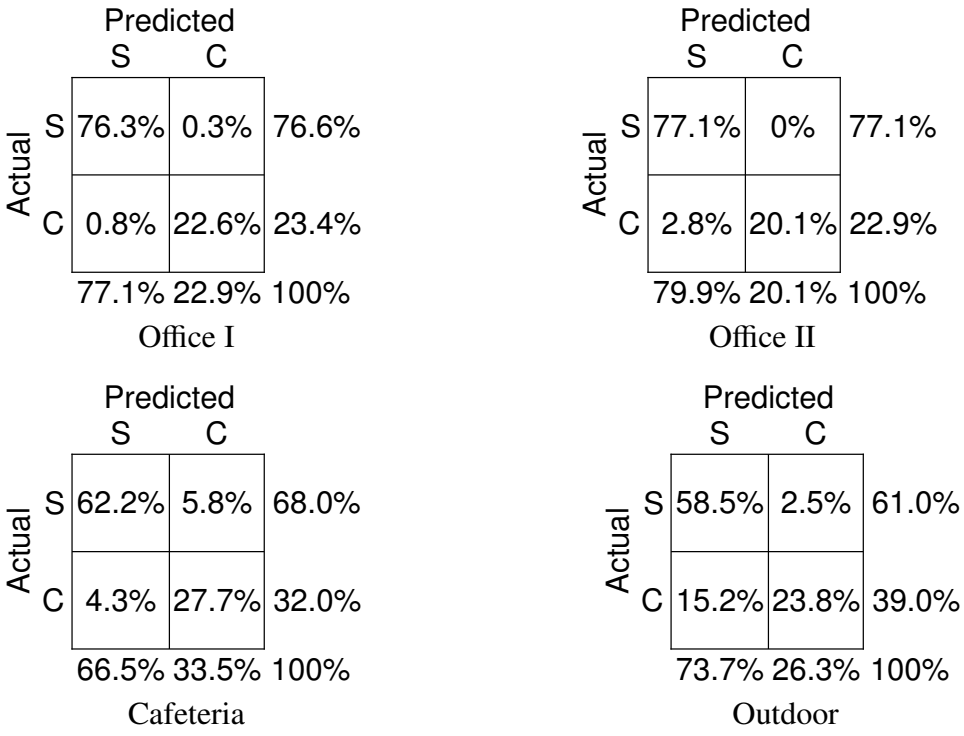


Figure 2.14: Confusion matrices detailing the classifier performance in the four environments. *S* is Sybil and *C* is conforming. Multiple tests were run in each environment, so mean percentages are shown instead of absolute counts.

Table 2.3: 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

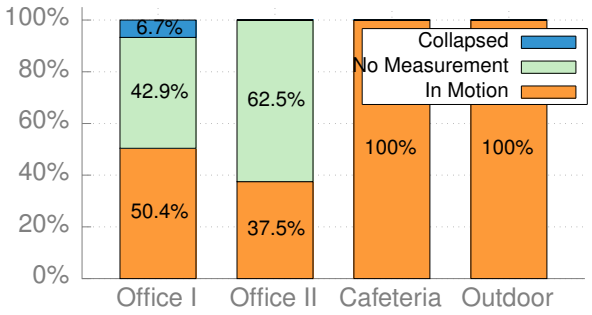


Figure 2.15: Relative frequencies of the three causes of false positives.

motion.

The outdoor experiment is an extreme case where we pay the cost of rejecting moving conforming nodes to defeat motion attacks. The result is acceptable because our goal is to produce a set of non-Sybil identities to be used safely by other protocols: accepting a swarm of moving Sybil identities is much worse than temporarily rejecting some conforming nodes that are currently moving.

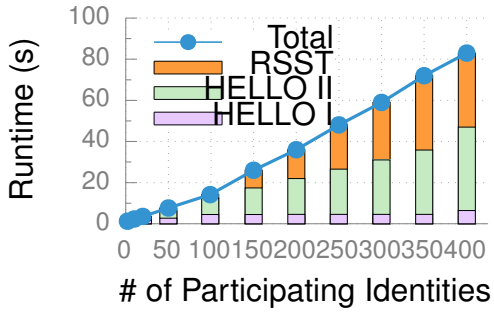
An identity is classified as Sybil for three reasons: it has similar signalprints to another, the initiator has too few RSSI reports to form a signalprint, or it is in motion. Figure 2.15 shows the relative prevalence of these three causes for falsely rejecting conforming nodes. Not surprisingly, the first cause—collapsing—is rare, occurring only in the first office environment. Missing RSSI reports is an issue only in the environments with significant obstructions (the indoor offices) and accounts for about half of these false rejections. In the open cafeteria and outdoor environments, all false rejections are due to participant motion.

2.9.3 Overhead Evaluation

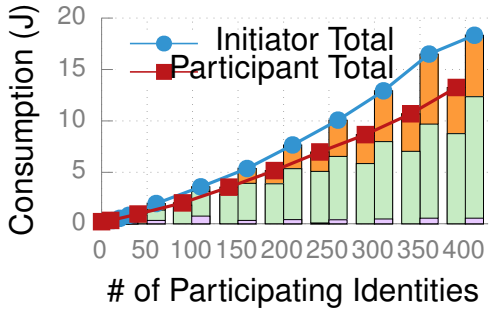
Figures 2.16a and 2.16b show the runtime and energy overhead for the Mason test collection phase, with the stacked bars separating the costs by sub-phase. The protocol runs infrequently (once every hour is often sufficient), so runtimes of 10–90 seconds are acceptable. Likewise, smartphone energy consumption is acceptable, with the extreme 18 J consumption for 400 identities representing 0.01% of the 17.500 J capacity of a typical smartphone battery.

Figure 2.17 show the classification phase overheads for 2–100 identities. Classification consumes much less energy than collection, so its overhead is 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

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



(a) Runtime in seconds of the collection phase.



(b) Energy consumption in joules of the collection phase.

Figure 2.16: Overhead of the collection phase. The stacked bars partition the cost among the participant collection (HELLO I), RSSI measurement (HELLO II), and RSSI observation exchange (RSST) steps.

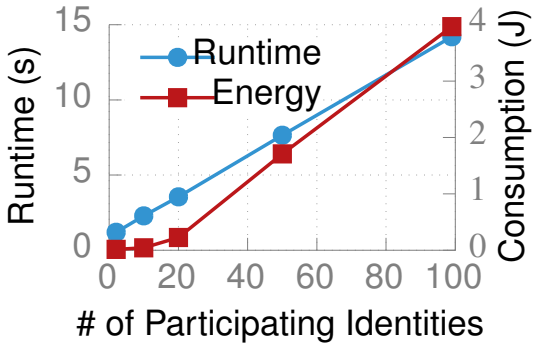


Figure 2.17: Overhead of the classification phases.

generally not reduce 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 in an 18 h period of use before recharging.

2.10 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 with the number of identities, $O(|I|^3)$. 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 a detailed treatment is beyond the scope of this work, we do note that 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 still 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.

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.

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 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 if their RSSI observations are correlated with the prior rounds (see Section 2.8.2).

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.

2.11 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 were eliminated due to motion.

CHAPTER III

MANES: A Mobile Ad Hoc Network Emulation System

3.1 Introduction

Mobile ad hoc and delay tolerant networks (MANETs and DTNs) have a number of advantages over infrastructure-based networks, including rapid low-cost deployment, continued operating in the face of natural events that would disable infrastructure, lack of susceptibility to attacks that rely on the hierarchical nature of real-world infrastructure networks, and resistance to censorship.

We argue that experimental deployments generally yield more accurate evaluation results for MANET and DTN protocols and applications than simulation. Such experiments allow evaluation under real conditions that cannot practically be recreated via simulation. Although sufficient in capturing first-order dynamics, simulation models may miss important aspects of reality, such as the effects of real-world wireless communication environments and the real-time interplay between application user behavior and network behavior. Trace-driven simulations can be used to take into account additional factors such as human motion patterns and network environments, but they also neglect the bidirectional influence between the users and the network.

Despite of its importance, MANET and DTN experimental deployment remain rare because it is laborious and expensive. A standard deployment usually involves recruiting human participants, equipping them with special-purpose devices with customized hardware and/or software, and setting up a mechanism for data collection and analysis. These tasks are often intractable for large deployments.

We argue that the quality of MANET and DTN research and evaluation results could be substantially improved if researchers had access to a low-cost method of deploying and evaluating their communication protocols applications on commodity mobile devices such as smartphones. Specifically, we propose supporting the implementation of these protocols and applications as commodity mobile applications that run on the personally owned mobile

devices of study participants. This approach not only eliminates the cost of changing system software or distributing dedicated experimental hardware, but also reduces the operating cost because existing cloud infrastructure, e.g., Google Play, becomes available for protocol installation and update. Another advantage of using the participants' own devices is the execution environment faithfully reflects the mobility and usage patterns of the users.

To facilitate this crowdsourcing approach to MANET and DTN experimentation, we aim to provide an integrated solution that guides researchers through the entire experimental deployment lifecycle. Specifically, this solution should satisfy the following requirements.

- It is an easily installed, low-overhead software-based solution capable of running on commodity mobile devices.
- It provides a general and convenient programming interface for wireless device-to-device communications, supporting 802.11 device-to-device connections, which have the best communication range and bandwidth among commodity mobile device ad hoc communication technologies.
- It supports real-time data collection and analysis during deployments.
- Its implementation is modular and easy to extend.

Our implementation is called MANES. MANES is an Android-based cloud solution. The client-side software provides convenient communication interfaces to the test protocols. It also constantly updates user data, e.g., location, connectivity, and packet transmission traces to the server. The server aggregates the data and provides the interface for real-time, network-wide event monitoring and logging. We take advantage of Android's existing infrastructure, i.e., Google Play, for protocol installation and update. MANES is provided as an open source software and users need to set up their own hosting servers for the deployment.

The main technical challenge in developing MANES was to provide a general and convenient programming interface for 802.11 device-to-device communications on commodity devices. Although 802.11 ad hoc mode is widely available on commodity wireless chips, Android does not expose the programming interface to application developers on un-rooted devices. Wi-Fi Direct provides device to device 802.11 connections, but current implementations are inappropriate for general use in MANETs and DTNs; for example, users must manually confirm first-time devices connections, making routing messages through third parties annoying to users [84].

In order to work around lack of general 802.11 device-to-device communication capabilities on commodity Android devices, MANES emulates them with the help of infrastructure

networks. Each transmitted packet is sent to a central server, which then replays the packet to the transmitter's one-hop 802.11 neighbors using a novel connectivity estimation algorithm integrated within MANES.

We observe that environmental information, e.g., Wi-Fi scan results and GPS readings, can be used to predict the direct connectivities between devices. Our prediction technique is based on a widely used heuristic in DTN communities which assumes that two devices are within Wi-Fi communication range if their scan results share common Wi-Fi Access Points [45, 46, 41, 52, 51, 48, 49]. We will refer to this technique as the "shared access point (SAP) heuristic" in the rest of the chapter. This technique tends to be overly optimistic: it is common for two devices scanning common APs to out of communication range with each other. We have developed a connectivity estimation technique based on geometric principles that is significantly more accurate.

In summary, this work makes the following contributions.

- It describes the design and implementation of MANES, a low-overhead, software based solution that facilitates experimental deployment to evaluate MANET and DTN protocols and applications using commodity mobile devices.
- It presents a new technique to predict pairwise 802.11 connectivities using environmental Wi-Fi scan results. We show that this prediction technique yields significantly more accurate results than widely used shared access point heuristic, according to an experimental evaluation with 17 users for more than 2,000 hours.
- This work is also the first we are aware of to experimentally evaluate the widely used shared access point connectivity estimation heuristic.

3.2 MANES Overview

In this section we first describe how to use MANES to help conducting a deployment. We then explain how MANES emulates 802.11 links and its limitations. Finally, the architecture and implementation details of MANES are described.

3.2.1 MANES Assisted Deployment

The deployment process with MANES is illustrated in Figure 3.1. The first step is to set up MANES servers and configure the MANES client to use them, so that user traces are transmitted to the correct backend server. The second step is to develop the test protocol as an Android application. To set up the development environment, the MANES client

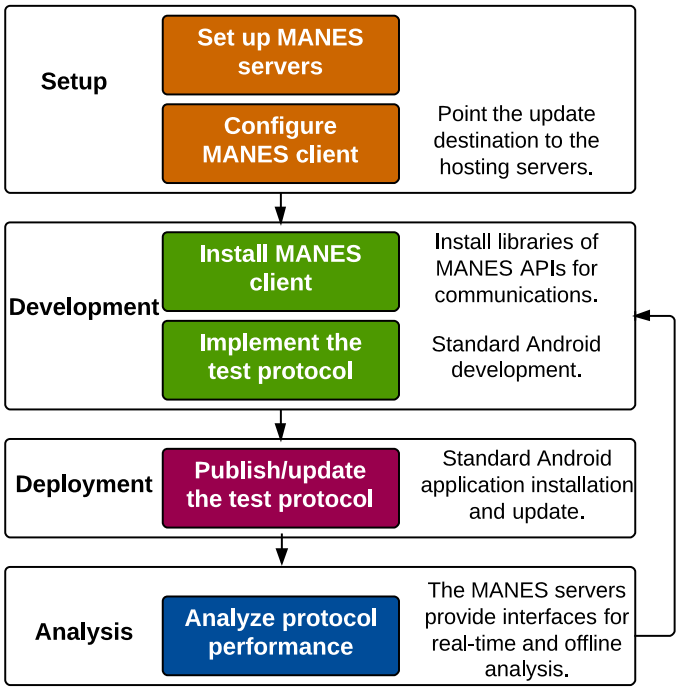


Figure 3.1: The deployment process with the help of MANES.

needs to be installed, which provides a library for device-to-device communication (see Section 3.2.1.1 for details). After protocol and application implementation and test, the third step is to publish the application to the participants. We recommend standard distribution channels, i.e., Google Play, to ease version management. Finally, during protocol execution, one may monitor performance by analyzing the frequently updated traces on the servers (see Section 3.2.1.1 for details). These traces are also permanently stored for offline analysis. In cases of bug patches or protocol updates, repeat steps 2 through 4 to install new versions of the application.

3.2.1.1 Communication APIs

MANES communication APIs provide data link layer service to higher-level protocols. MANES is thus mainly useful for protocol developers working at the network layer and beyond. Unlike infrastructure-based network emulation systems, MANES do not provide services for the network or the transport layer; protocol designs at these layers of MANETs and DTNs are still subjects of active research—MANES has the goal of making accurate evaluation of these designs practical.

MANES communication APIs provide a straight-forward data link service. Hosts may transmit and receive packets to and from other hosts with direct Wi-Fi connectivities.

We observe that because collision domains in MANETs and DTNs are usually small, the advantages of supporting link layer unicast do not justify the additional engineering complexity. We thus remove link level addressing mechanisms (e.g., MAC addresses) and broadcast packets. Higher-level protocols must implement addressing logic.

MANES communication APIs serve higher-level protocol's data in units of 802.11 MAC Service Data Units [85]. It does not provide fragmentation/aggregation services.

To use MANES APIs each protocol/application has to register an unique protocol id with the system, which is used to demultiplex incoming packets [86]. This design is in compliance with mainstream Internet networking systems where layer-3 protocols register protocol IDs with network interfaces [86].

MANES provide two methods in particular, "send" and "receive", which are similar to commonly understood UDP datagram socket APIs. "send" immediately delivers the packet to the wireless card and returns. "receive" is a blocking call that returns either with incoming packets (of the caller's protocol ID), or after a specified timeout period. We recommend using a dedicated working thread for receiving.

3.2.1.2 User Traces

Currently MANES gathers three traces for each client: the location trace, the topology trace, and the packet transmission trace. As soon as they are available, new entries in each of these are transmitted to MANES servers. The location trace contains the client's periodic location updates, including Wi-Fi scan results and GPS readings. The topology trace contains the client's neighbor list, also updated periodically. The transmission trace records each transmission from the client, including the time, raw bytes, and the corresponding receivers.

The traces are stored as files on the MANES servers, with a dedicated subdirectory for each client. For example, "topology/1/" stores the current topology trace of Client #1 and "packets/5/" stores the current transmission trace of Client #5. The traces are wrapped up and moved to permanent storages on a daily basis, e.g., to Amazon Simple Storage Service [87]. The real-time and historical traces provide complete information of the current and past events of the deployment.

3.2.2 802.11 Link Emulation

Emulating 802.11 links requires understanding their real-world behavior. A 802.11 link is mainly described by two factors, the data rate and the Packet Reception Rate (PRR). 802.11 supports multiple data rates ranging from 1 Mbps in 802.11b to 6.75 Gbps in 802.11ad,

and it relies on a complex rate selection mechanism to select the most appropriate data rate according to the current Signal to Noise Ratio (SNR), so that a nearly 100% PRR is achieved [88]. Given a SNR value, the higher the data rate, the smaller the PRR is [88].

Recall that MANES emulates a 802.11 link using an Internet communication channel passing through a backend server. Because the data rate of this channel 1) is under the influence of many factors out of our control and 2) is only comparable to the low data rates of 802.11, e.g., less than 10 Mbps [89], it is impractical to use this channel to accurately emulate the rate selection behavior of 802.11 links.

Therefore, MANES only emulates 802.11 links operating at fixed data rates, e.g., 1 Mbps in 802.11b and 6 Mbps in 802.11g. Specifically, MANES estimates the PRR assuming a fixed low data rate and relays packets accordingly. This poses a problem that when application data saturates the MANES communication channel. In this case, the actual bandwidth does not accurately match that of the true 802.11 links. It does not affect low bandwidth applications whose data rate is far below a few Mbps, e.g., text messaging and news distribution. However, it negatively affects the performance of latency-sensitive high-bandwidth applications such as online gaming, VoIP, and streaming. We note that due to latency variability, many MANETS and DTNs are inappropriate for such applications.

Another side effect of MANES' emulation methodology is the prolonged link delay resulting from traversing the Internet to and from MANES servers. We analyze the impact of this delay in Section 3.4.

3.2.3 MANES Architecture

MANES continuously updates and maintains the real-time topology of an entire MANET or DTN and use this information to allow accurate packet delivery routes and delays. Figure 3.2 illustrates the architecture of the MANES client-server system. On the client side, the packet I/O maintains the actual communication channels to and from the MANES servers and provides the illusion of device-to-device communications to higher-level protocols. The location tracker constantly monitors and reports link quality related information, i.e., Wi-Fi scan results and GPS readings, to the MANES servers. On the server side, the topology estimator reevaluate the network topology upon each location update, and refreshes the topology database accordingly, so that the topology database always stores the latest neighbors of each node. When a transmitted packet arrives from the clients, the packet relayer consults the topology database and relays the packet to the sender's neighbors. The client information database contains registration information of each client and is used for authentication purposes.

We now discuss a few important design details.

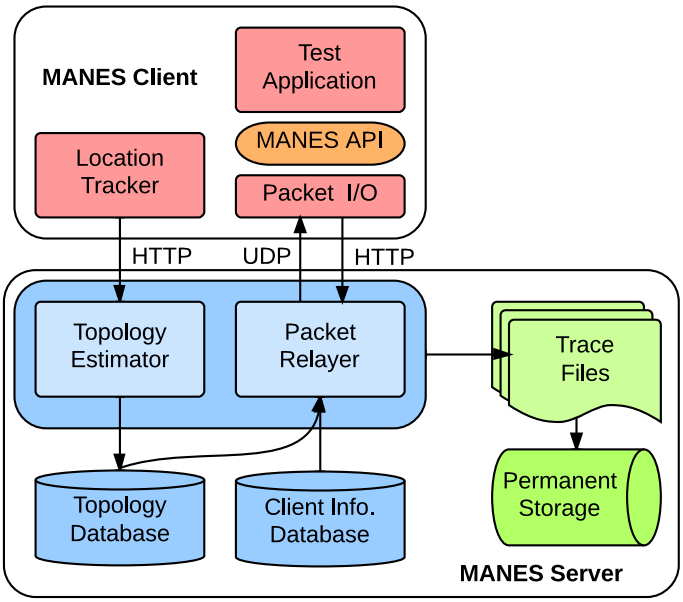


Figure 3.2: MANES architecture.

3.2.3.1 Scalable Architecture

To accommodate varying deployment scales, an important design goal is to support convenient addition and removal of server instances. We achieve this goal by storing shared global states, i.e., the current topology of the network, in a distributed database. Since each server instance needs only the global information to perform its logic and has no inter-dependency or direct data exchanges between each other, scaling is accomplished by adding or killing server instances. For this approach to work, the database must also be scalable. MANES uses the open source Project Voldemort key-value storage system [90].

3.2.3.2 Real Time Topology Estimation

A naïve way to conduct topology estimation is to recalculate the connectivities between the sender and all the other clients upon each location update. This O(n) algorithm does not scale well. We reduce the complexity by only considering the sender's potential neighbors, i.e., the clients with common APs or within reasonable GPS distances (when available) to the sender. Each new evaluation updates the sender's neighbor list, as well as those of its neighbors, so that the estimated connectivities are symmetrical. Specifically, the sender is added to the neighbor lists of its new neighbors and removed from the neighbor lists of its old neighbors, i.e., those that are no longer connected according to the new evaluation.

In addition to location updates, another trigger for topology reevaluation is location expiration, i.e., a client has not updated its location for so long a time, e.g., 5 minutes, that

we consider the client to be inactive. In this case we simply remove the client from all its latest neighbors' neighbor lists and clear its own neighbor list.

3.2.3.3 Link Level Delay Optimization

Emulation can increase the latency of device-to-device communication because MANES packets actually travel through the Internet and are processed by the MANES servers. The server code was optimized reduce this delay. In addition we use a notification mechanism with the smallest delay, the "push" mechanism, which requires keeping the server-to-client paths constantly open. Specifically, each client periodically probes the server, e.g., every 45 seconds, to refresh the firewalls and Network Address Translators along the path.

3.2.3.4 Security and Privacy

Security is an important concern because MANES uploads fine-grained location information to the servers: researchers using MANES are ethically responsible for informing study participants that they will have access to location data. There is also a risk of third-party attackers eavesdropping on users' locations and/or send fake location updates to manipulate messages deliveries. Therefore, each location update is automatically encrypted and signed by the sender, the parameters for which are negotiated during the initial user registration.

3.3 802.11 Connectivity Estimation

The main technical challenge of emulating 802.11 links is to estimate whether two devices are connected without establishing an actual communication channel between them. Previous research demonstrated that Received Signal Strength Indicator (RSSI) values are well correlated with signal-to-noise ratio (SNR) and thus a practical predictor of connectivities [88]. Our solution focuses on utilizing this information to predict 802.11 connectivities. We assume a fixed data rate, 6 Mbps, the lowest data rate supported by OFDM [85].

For devices supporting Wi-Fi Direct (Android 4.0 or later) we propose to directly measure the 802.11 channels via management frames. Specifically, Wi-Fi Direct devices operate in autonomous mode as group owners. Because group owners behave as normal APs and periodically broadcast beacons [85], other devices, including ones without Wi-Fi Direct, can measure these beacons through normal Wi-Fi scans. Since beacons are sent over low data rate 802.11 channels (6 Mbps OFDM) [85], scan results are direct channel

measurements; a scanning device is connected with a group owner if the device scans the group owner's network and not connected otherwise.

For devices without Wi-Fi Direct support, we observe that environmental information, e.g., Wi-Fi scan results, can be used to estimate pairwise RSSIs and thus the direct connectivities between devices. The DTN community widely uses the shared access point heuristic (SAP), which considers two devices to be within communication range if their scan results have share APs [45, 46]. We improve upon the SAP heuristic by using geometric relationships to estimate pairwise RSSIs.

The rest of of this section describe the Wi-Fi scan based connectivity estimation algorithm and its experimental evaluation.

3.3.1 Wi-Fi Based Connectivity Estimation

The algorithm involves two steps. First the direct RSSI between two devices is estimated based on their RSSI measurements of nearby APs. Then the estimated RSSI is compared to a predetermined threshold to decide whether the two devices are connected. Before explaining RSSI estimation in detail, we first introduce the relationship between RSSI and 802.11 link connectivity.

3.3.1.1 RSSI and Connectivity

Previous research has shown, both theoretically and experimentally, that for a given 802.11 data rate, the connection probability of two devices, quantified as Packet Reception Rate (PRR), increases with SNR. The function has a sharp transition region within a couple dBm during which the PRR increases quickly from 0 to 1 [86]. The same phenomenon is observed for the PRR to RSSI curve, with a wider transition region, however, as RSSIs do not perfectly predict SNRs [91].

Our experiment conducted with commodity Android devices (HTC Passion with Broadcom 802.11a/b/g/n wireless card BCM4329) in various locations of a campus building also confirms the result. As shown in Figure 3.3, the curve has a short transition region from -95 dBm to -90 dBm. Note that the variations in the [-90, -70] region are probably due to varying noise levels that are not well captured by RSSIs.

Because 1) the transition region is very small and 2) PRRs within that region are very sensitive to RSSI variations caused by estimation errors, we do not attempt to model this region in MANES. Instead, we use a step function with a heuristic threshold.

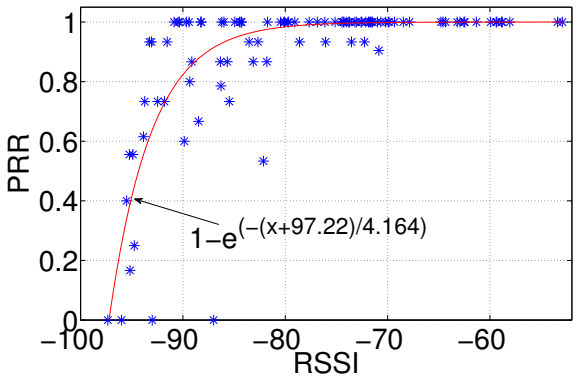
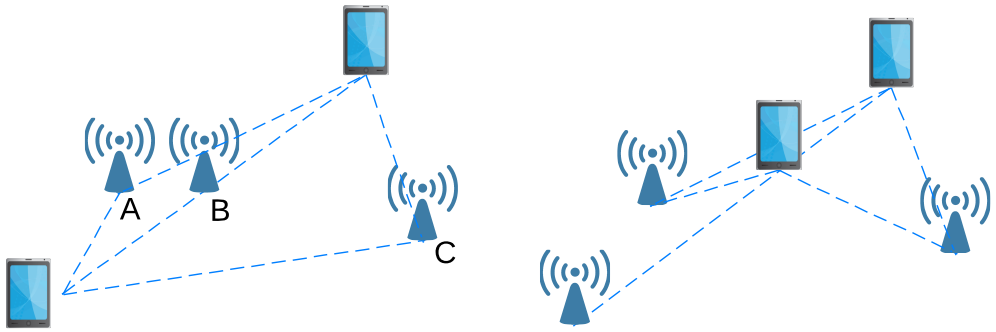


Figure 3.3: The PRR-RSSI relationship as measured by our own experiments.



(a) No underestimation. (b) Large underestimation.

Figure 3.4: Examples cases of comparative locations between APs and the device-to-device path.

3.3.1.2 RSSI Estimation

The RSSI estimation problem is closely related to the intensively researched Wi-Fi localization problem. RSSI can be estimated with some accuracy once device locations are known. Solutions to the Wi-Fi localization problem can be divided into two classes. The model-based solution relies on Wi-Fi propagation models to triangulate device locations. Its accuracy is often compromised by random wireless channel variations [92]. The measurement-based solution characterizes wireless channels more accurately, but requires labor-intensive site surveys [73]. Recently proposed methods use additional sensors, e.g., accelerometers and GPS, to increase accuracy and reduce pre-characterization efforts [93].

To simplify deployment for researchers using MANES, we developed a model-based method that does not require pre-characterization or additional sensors. As shown in Figure 3.4a, two devices measure RSSIs of multiple common APs. Assuming a sufficient spatial density of APs, some of them will be located close to the device-to-device path, e.g., the AP labeled B in Figure 3.4a. In this case, the device-to-device distance is the sum of

device-to-AP distances, which makes it easy to calculate the device-to-device RSSI from the device-to-AP RSSIs. That said, since multiple common APs are measured, we still need to determine which ones sit on the direct path. As shown in Figure 3.4a, Euclidian geometry implies that the APs closes to the direct path have the shortest sum of device-to-AP distances, which can be inferred from device-to-AP RSSIs. Although inferring distance from RSSI introduces error, we will later demonstrate that the proposed approach significantly improves estimation accuracy compared with the current state of the art.

We now derive the mathematical expression of this method (which we refer to as the MANES method in the rest of the chapter). Recall that the path-loss model describes the signal strength P between two devices as

$$P = P_0 - 10r \cdot \log(d), \tag{3.1}$$

where d denotes distance, P_0 denotes transmitter power, and r is the path-loss exponent [94]. Assuming the i-th AP sits on the direct device-to-device path, we can express the device-to-device signal strength as a function of the two device-to-AP signal strengths P_{i1} and P_{i2},

$$P_i = -10r \cdot \log(10^{\frac{P_{i1}}{10r}} + 10^{\frac{P_{i2}}{10r}}). \tag{3.2}$$

Considering all the N common APs, the estimated signal strength $\tilde{P}$ is

$$\tilde{P} = \max(P_1, P_2, \cdots, P_N). \tag{3.3}$$

Note that this method tends to underestimate, because the sum of device-to-AP distances is always larger that the actual device-to-device distance, as APs are always a non-negative distance away from the direct path. For example. the signal strength in Figure 3.4b is largely underestimated because all the APs are far away for the direct path. Interestingly, this inherent error varies with device-to-device distances; for a given AP density, the larger the device-to-device distance, the smaller the estimation error, because it is more likely for a long path to have a nearby AP. We observe that this property well serves our specific problem. As shown in Figure 3.3, the critical RSSI region for connectivity estimation is below -90 dBm, which corresponds to large device-to-device distances.

3.3.2 Experimental Evaluation

We evaluated the connectivity estimation algorithm using connectivity traces from 17 participants with a total duration of more than 2,000 hours. These participants carries smartphones running custom system software allowing us to gather ground truth connectivity

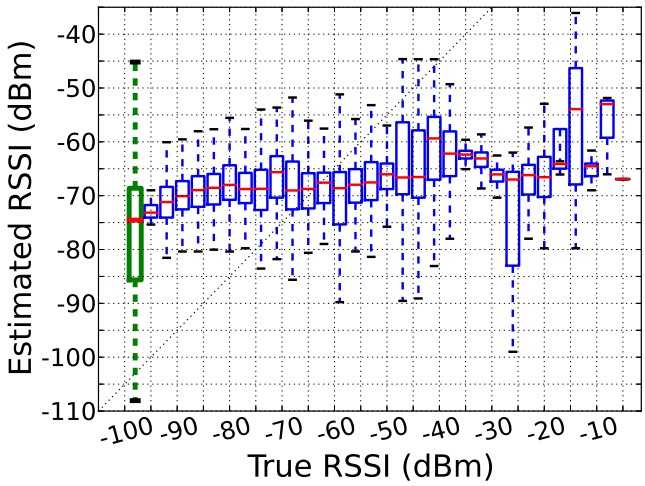


Figure 3.5: RSSI estimation performance of MANES' method.

data. We also evaluated the widely used SAP heuristic, i.e., two devices are in connection as long as they scan common APs, as a reference point. We report False Positive Rate (FPR) and True Positive Rate (TPR) numbers. Since the eventual use of these connectivity estimation algorithms is to produce MANET and DTN instances for protocol evaluations, we also study the resulting network traces. Specifically, we produced DTN contact traces from pairwise connectivity estimations and compare their statistics with ground truth statistics.. Finally, to have a direct understanding of how these traces influence protocol evaluations, we use them to drive simulations of a simple flooding protocol and compare its performance using different traces.

3.3.2.1 Trace Collection

We gathered connectivity traces from 17 participants for more than 2,000 hours. The experimental platform is a Galaxy Nexus with Broadcom 802.11a/b/g/n wireless card BCM4330, running a background service that periodically scans Wi-Fi networks every 20 seconds. The device also has a cell phone plan (T-Mobile) mainly for time synchronization.

Ground truth connectivities were gathered in the following way. Each device operates as an autonomous Wi-Fi Direct group owner, constantly broadcasting SSIDs. If one device scans another device's network the two are connected and otherwise disconnected.

To gather connectivity data, we chose participants in groups that have co-location opportunities. There are 4 groups in total, a 3-person group that go to a two day hiking trip together, a 5-person group that work in two adjacent offices, a 4-person group that reside in the same house, and a 4-person group that attend a same graduate school class. The

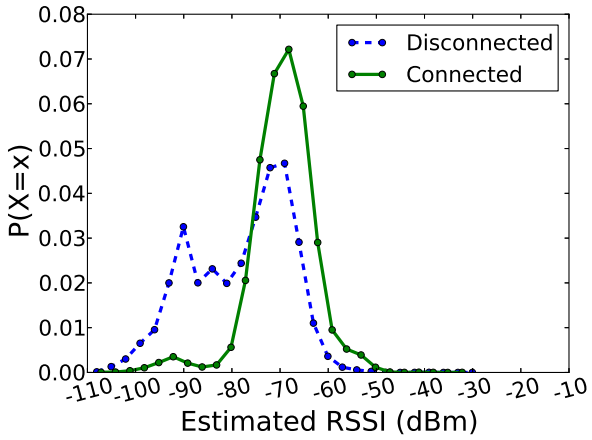


Figure 3.6: Distributions of estimated RSSIs for the connected and disconnected device pairs.

participants were asked to carry the experiment devices as their primary smartphones for a week.

3.3.2.2 RSSI and Connectivity Estimation

Figure 3.5 shows the performance of MANES' RSSI estimation algorithm. There is a clear correlation between the true and estimated values. However, our algorithm tends to underestimate when the true RSSIs are large and overestimate when they are small. The first case is expected as the algorithm has an inherent tendency to underestimate. For the second case, a closer look into these data samples suggests a possible explanation; some APs have significantly larger transmit power than others and estimations based on their measurements are also larger. We are in the process of developing an AP calibration method to solve this problem. Despite these two sources of inaccuracy, MANES still produces results significantly more accurate results that other state-of-the art connectivity estimation techniques.

We ultimately care about the separation power of the estimated RSSIs between the connected and disconnected cases. Figure 3.6 shows that the distributions of estimated RSSIs for the two cases are well separated. We use a heuristic RSSI threshold of -85 dBm for classification. Table 3.1 shows the resulting False Positive and True Positive Rates for the MANES and the SAP method. As expected, SAP has a high FPR because two devices scanning common APs are not necessarily connected. The MANES method reduces the FPR by distinguishing disconnected cases using RSSI estimations.

Table 3.1: Classification Performance of Pairwise Connectivities

	FPR	TPR
SAP	0.054	0.998
MANES	0.023	0.995

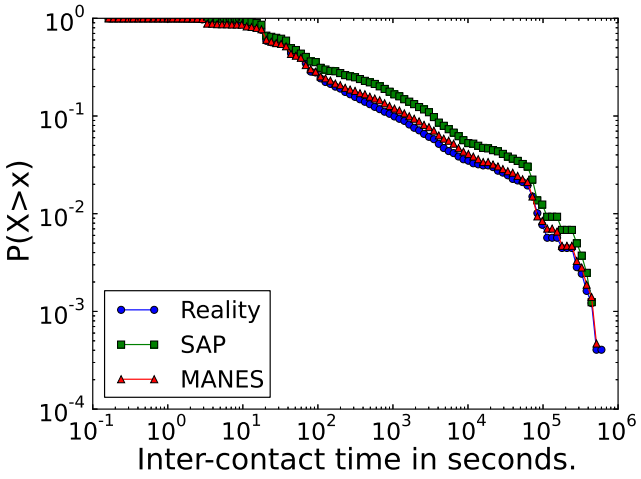


Figure 3.7: CCDF of inter-contact times of the real, MANES produced, and SAP produced contact traces.

3.3.2.3 DTN Statistics

To understand how pairwise connectivities influence the resulting network instances we extract contact traces from connectivity estimations. Contact traces are used to drive DTN protocol simulations and are thus critical to their evaluation. As a matter of fact, a large body of DTN protocols are evaluated using estimated contact traces generated from Wi-Fi scan traces by the SAP heuristic. The Dartmouth [45] and UCSD traces [46] are commonly used in this way.

We show two metrics of the contact traces that are considered critical to the performance of DTN protocols: the inter-contact time distribution and contact-duration distribution [41].

Inter-contact time is the time separation between two consecutive contacts of a pair of devices. It represents how frequently a device meets other devices. As shown in Figure 3.7, the distribution produced by the MANES method is closer to reality, while both methods are more likely to have long inter-contact time than the reality. A likely explanation is false positive connectivities fill in the small disconnected “gap”s between contacts and in effect remove these short inter-contacts. MANES performs better because it is less likely to have false positives.

As shown in Figure 3.8, the same phenomenon is observed for the contact duration

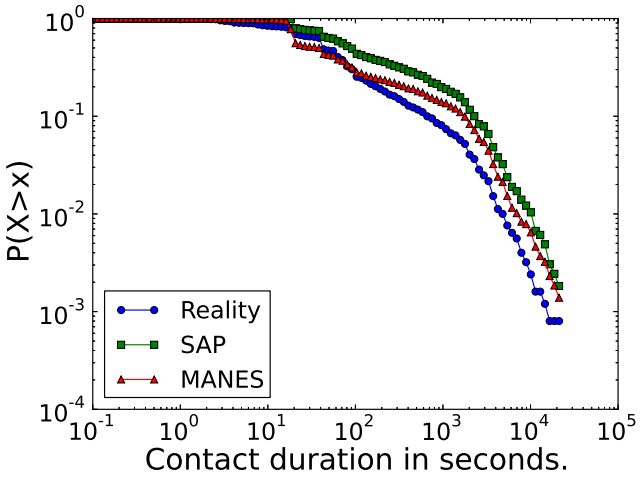


Figure 3.8: CCDF of contact durations of the real, MANES produced, and SAP produced contact traces.

distributions; both methods are more likely to have long contact durations than the reality, while the problem is less severe for MANES. A likely explanation is false positive connectivities extend the duration of each contact. Also, two short contacts separated by a small disconnection "gap" may be merged into one long contact by false positive connectivities.

3.3.2.4 DTN Protocol Simulation

Finally, we conduct simulations of a simple flooding protocol using the real, MANES produced and SAP produced contact traces, and compare the resulting protocol performance. During the simulation every node initiates the flooding process every three hours. For each flooding process we record the percentage of the network reached each hour.

Figure 3.9 shows the medium flooding process for all three traces. Both traces propagate messages faster than reality, while the MANES traces are much more closer to reality. For example, to deliver a message to the whole network, SAP is ~24 hours (34%) faster than the reality, while MANES is only ~5 hours (7%) faster. Note that the network sizes of these traces are less than 5 at any point of time, as our largest participant group is of 5 persons. We thus expect a much larger performance mismatch for larger networks.

3.3.3 Conclusions and suggestions

The SAP method is widely used in DTN communities to produce contact traces and drive protocol evaluations [41, 52, 51, 48, 49]. This experimental evaluation is the first attempt, to our knowledge, to evaluate the SAP method and study its influences over protocol

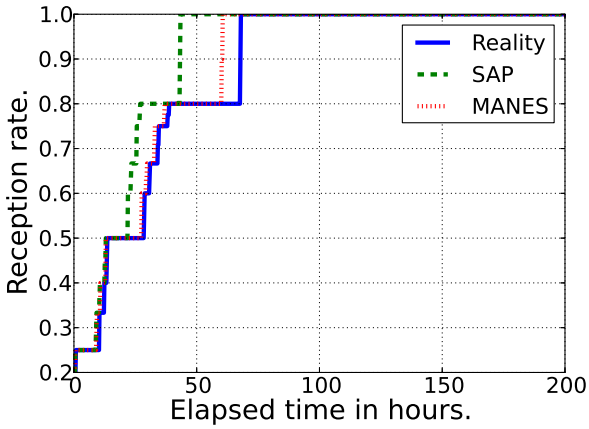


Figure 3.9: Simulation performance of a flooding protocol using the real, MANES produced, and SAP produced contact traces.

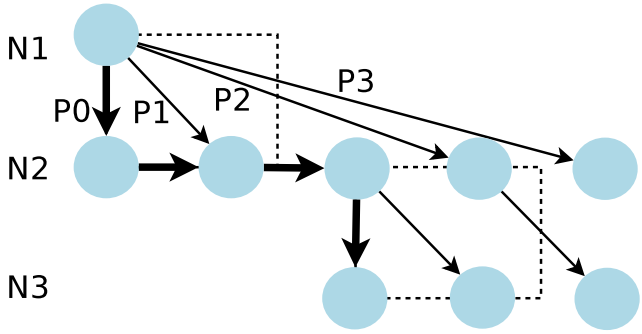


Figure 3.10: The effect of link delays.

evaluations. Our evaluation results show that it is promising to use environmental Wi-Fi RSSI measurements to conduct pairwise connectivity estimations. However, we found a significant mismatch between protocol performances estimated using the SAP connectivity traces and the real traces, and also found that the MANES connectivity estimation method yields results significantly closer to reality. It is our goal to raise awareness of this problem and provide a way for researchers to significantly improve the accuracies of their evaluation results in the future.

3.4 Emulation Overhead of Link Delays

Link delays influence protocol executions in three different ways, judged by whether the original packet delivery path and the overall delay over that path are changed by the link delay. Figure 3.10 illustrates these three cases with a simple example where device N_1 tries to send a packet to N_3 through N_2. Each row represents the states of a device over time. A

dashed rectangle between two rows represents a period of time when the two corresponding devices are in contact. N_1 first meets N_2 and delivers the packet immediately. After some time N_2 meets N_3 and is kept in contact for a while. Path P_0 (marked with the thickest arrows) shows the original packet delivery path with no link delays.

The three cases follow. Link delay may have no influence on either the packet delivery path or the overall delay, as shown by P_1 with a link delay of D_1. It may not influence the packet delivery path, but prolongs the overall delay, as shown by P_2 with a link delay of D_2. It may change the packet deliver path (and therefore the overall delay), as shown by P_3 with a link delay of D_3.

In P_1 the link delay is "buffered" by the wait time before encountering the next contact and therefore has no influence over either the packet delivery path or the overall delay. This happens in networks with rare encounters and disrupted paths, i.e., *Delay Tolerant Networks (DTNs)*.

In P_2 the packet delivery path is not changed because the topology remains constant during the delay interval (although the packet arrives at N_2 at a delayed time, N_2 is still able to deliver it to N_3 because they kept in contact during this time). However, the overall delay adds up as the number of hops increases. This happens in networks with comparably stable topologies, e.g., static networks.

In P_3 N_2 could not deliver the packet to N_3 because it has already lost contact with N_3 when the packet arrives. This happens in networks with comparatively fast changing topologies, e.g., vehicle ad hoc networks.

Specifically, MANES introduces less than 200 ms delay on each emulated (potentially multi-hop) link, most of which spent traversing the Internet to and from the MANES server). For DTNs (e.g., sparse networks with human-carried devices) this comparatively short delay has no influences over protocol execution because it is often "buffered" and annihilated by inter-contact gaps, which ranges from minutes to days. Therefore MANES can be used to evaluate DTNs. MANES cannot evaluate networks that are so highly dynamic that topology changes commonly occur every 200 ms. However, it is appropriate for less dynamic MANETs and DTNs, e.g., most network composed of human-carried mobile devices.

3.4.1 Maximum Tolerable Link Delay

To help users decide whether MANES can be used to evaluate their target networks, we propose the notion of *maximum tolerable link delay* to quantify a network's sensitivity level to link level delays. Networks with maximum tolerable link delays comparable to or smaller than a few hundred milliseconds cannot be evaluated by MANES. Users can estimate this metric for their target networks via simulations with realistic mobility models, for example.

Algorithm 3 Calculate the maximum tolerable link delay T_{tor} of a path.

```
cnum ← the total number of contacts in the path
T_tor ← ∞, i ← 1
while i ≤ cnum do
    C_i ← the i-th contact in the path
    T_i ← the packet arrival time during C_i
    j ← i + 1
    while j ≤ cnum do
        C_j ← the j-th contact in the path
        T_j ← the end time of C_j
        T ← (T_j − T_i)/(j − i)
        if T < T_tor then
            T_tor ← T
        end if
        j ← j + 1
    end while
end while
```

The *maximum tolerable link delay* over a packet delivery path is the link delay beyond which the packet can no longer be delivered over that path. For example, for the $N_1 \rightarrow N_2 \rightarrow N_3$ path in Figure 3.10, delays D_1 and D_2 (which produce P_1 and P_2 respectively) are tolerable, as the packet can still be delivered over that path. D_3 (which produces P_3) is not tolerable, as the packet can no longer reach N_3 via N_2. This metric is related to the time continuity (i.e., how often the path is interrupted by waiting for the next contact) and the stability (i.e., how long each contact lasts) of the path. The maximum tolerable link delay of a network is an accumulated statistic for all the shortest paths in that network.

Algorithm 3 presents the procedure for calculating the maximum tolerable delay of a given path. Note that we model a path as a sequence of contacts, and each packet delivery happens during a contact. For example, the $N_1 \rightarrow N_2 \rightarrow N_3$ path composes of two contacts, $N_1 \leftrightarrow N_2$ and $N_2 \leftrightarrow N_3$.

We generated the maximum tolerable link delay distribution of a realistic DTN composed of human carried devices. The traces are generated from the association histories of all the Wi-Fi devices in Dartmouth College from 2003 to 2004 [45]. We believe the resulting network is representative of DTNs with human carried devices. Figure 3.11 shows the distribution, whose 95th percentile is 571 ms, which is far greater than the link delay introduced by MANES. This indicates that MANES can be used to evaluate DTNs with human-carried devices. Figure 3.12 shows that the maximum tolerable link delay decreases with the device density of the network. Dartmouth traces (the point marked as "Dartmouth College") has a very high device density (more than 4,000 devices in a 0.6 km^2 area),

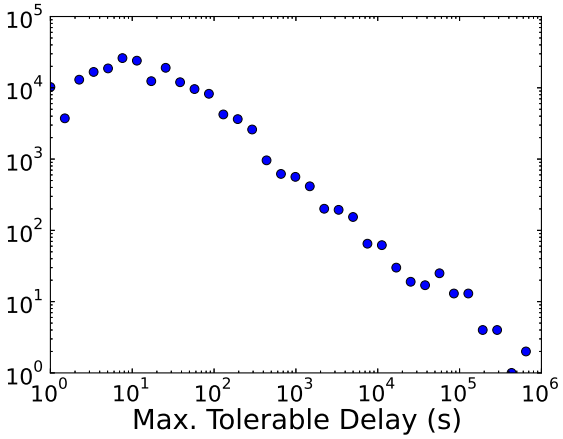


Figure 3.11: The maximum tolerable link delay distribution of Dartmouth traces.

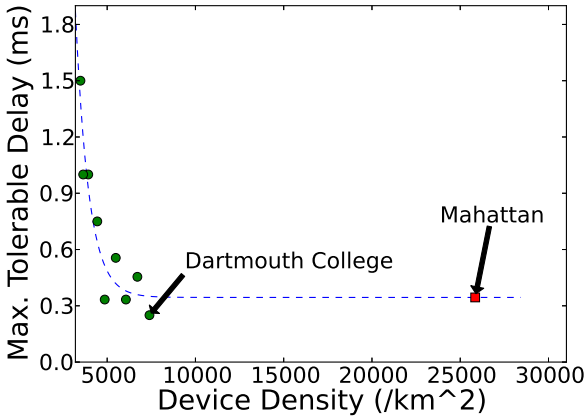


Figure 3.12: The 99th percentile of maximum tolerable link delays vs. device density.

meaning more than 70% of the student body participated in that network. We expect sparser device densities for most MANET and DTN studies and therefore higher tolerance to link delays, as shown by the points with smaller densities, which were generated by randomly removing devices from Dartmouth traces.

3.5 Deployment Experiences with MANES

We used MANES to conduct a large scale deployment of a DTN application, involving around 300 participants. The application is called 1am, a censorship-resistant news distribution service built on a DTNs composed of human-carried devices. MANES allows us to connect important information about user movement and message propagation. We will describe in Chapter IV and Chapter V the details and discoveries of this deployment.

3.6 Conclusion

We developed and implemented MANES, an emulation system to enable MANET and DTN researchers to easily conduct experimental deployments of their protocols and applications. We have demonstrated improved network topology estimation accuracy relative to the widely used shared access point heuristic without imposing precharacterization burdens on researchers.

CHAPTER IV

Campus Delay Tolerant Networks With One Hundred Thousand Participants: Trace Collection

4.1 Introduction

In recent years with the emergence and popularization of high performance, multi-functional mobile computing devices, e.g., smartphones and laptops, infrastructureless networks composed of these devices are attracting increasing attention from not only researchers and developers [32, 41, 42, 33, 43], but also end users. In 2014, a local mesh networking chat program called FireChat experienced explosive downloads from protesters of the "Umbrella Movement" in Hong Kong, due to the threat of potential denial of service attacks on the Internet-based Instagram [34].

These self-organizing, device-to-device networks are usually modeled as Delay Tolerant Networks (DTNs). Since node-to-node communication depends on short range radio, e.g., 802.11 or Bluetooth, end-to-end connectivity often does not exist and human movements play a major role in ferrying information, causing comparatively long delays.

Because of their natural resistance to surveillance and censorship, DTNs have the potential to support secure, censorship-resistant communication. It is expensive to control these networks, as it requires controlling a large portion of the participating devices privately owned by hundreds of thousands of people. It is even more expensive to shut them down, as any two devices could form a network and start information exchange. Because DTNs experience comparatively long delays, e.g., hours instead of milliseconds, we envision using them to support local, latency insensitive information exchange, e.g., within a university or company campus, a town, or a small city with hundreds of thousands of people.

The foremost challenge in designing practical DTN systems capable of serving a local community is our lack of understanding of large scale DTNs, in particular their capacities in terms of message delivery performance, robustness to node failures, and energy overhead.

This lack of understanding is rooted in the lack of fine granularity, large scale device connectivity (or contact) traces. It is difficult to collect large scale device contact traces, because it requires recruiting a large number of participants. We cannot piggyback the data collection on mobile platforms with established user groups, either, as there are no popular platforms or applications that actively utilize wireless device-to-device communication. The largest existing trace to our knowledge involves 22,341 students in a university campus, and is inferred from these students' course schedules [47]. The granularity of this trace is likely much coarser than reality, as all devices in the same class are assumed to be able to communicate during the entire course of the class. In addition, this trace considers only contact opportunities resulting from sitting in same classes, overlooking a significant number of other opportunities, e.g., residing in the same dormitory, or having lunch in the same dining hall.

We attempted two methods to collect fine granularity, large scale mobile device connectivity traces. The first method is active installation of measurement software on volunteer users' personally owned smartphones. We developed a prototype microblogging application on Android, named *1am* [60], and made it available to volunteers in the University of Michigan campus. Over a period of seven months in 2013, 1am had 291 users, with 111 users during a month of high activity on which our analysis focuses. We collected fine granularity, device-to-device wireless connectivity traces from these volunteers through continuous and frequent connectivity sampling, on average every 115 seconds. We refer to these traces as "the 1am dataset" in the rest of the thesis.

The second method is passive collection of WiFi association records, which was first used by Song and Kotz to collect large-scale device contact traces at Dartmouth College [95]. They obtained WiFi association record from their campus WiFi network's operator, and use it to infer direct 802.11 contacts; two devices associated with the same access point simultaneously are considered to be in contact. As campus WiFi network tends to cover popular locations with most human activities, this method records a significant portion of on-campus contacts. Its coverage is even better nowadays with the ever-expanding WiFi coverage and ever-increasing adoption rate of smartphones on university campuses.

We use the same WiFi-based method to gather contact traces on the University of Michigan campus, involving 119,055 mobile devices for a month. This number is about twice the U-M population, indicating a high coverage ratio of the campus population. This is also the largest scale human contact trace, to the best of our knowledge. We refer to these traces as "the U-M dataset".

To understand its underlying network structure, we construct a connectivity graph of human contacts involving 119,055 mobile devices from the U-M dataset. We study basic

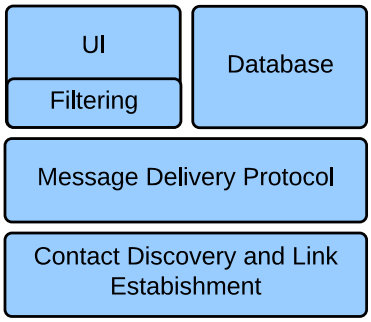


Figure 4.1: 1am architecture.

properties of this graph, reporting significantly higher node degrees (2863), shorter path lengths (2.3), and higher cluster coefficient (0.357) than popular online social networks. We show that its degree distribution has an exponential, instead of power-law tail. We also report that the human contact network shows strong "assortative mixing".

The remaining parts of this chapter focus on describing the trace collection methodology and basic statistics of these two datasets. It gives an overview of participants' motion and contact patterns, and provides background for understanding the performance numbers in Chapter V. Specifically, Section 4.2 and Section 4.3 describe the 1am dataset. Section 4.4 describes the U-M dataset. In Section 4.5 we study basic properties of the static human contact graph constructed from the U-M dataset. The detailed analysis on message delivery performance, robustness, and energy overhead of these DTNs is described in Chapter V.

4.2 1am: Microblogging on DTNs

We developed a prototype of *1am*, an infrastructureless DTN-based microblogging application, and provided it to volunteers in a college town with a student and faculty population of 50,000. The application was instrumented (with knowledge of the study participants) to gather mobility and wireless connectivity data. This section describes the design of the 1am microblogging application, providing background and context to the subsequent performance analysis sections in Chapter V.

4.2.1 Overview

1am is designed to functionality similar to Twitter. Posts are public text messages containing up to 514 characters. Users can post new messages, make comments, and retransmit (analogous to retweet) messages and comments from others. Its main user interface (UI) displays a timeline of incoming messages.

We now describe some of 1am's characteristics and features.

Identity Management: 1am has no central registry that might be used to enable blocking, censorship, or surveillance attacks; it uses a fully distributed design. Users are uniquely identified by self-generated public keys, created locally during application installation. In other words, the public key is the identity, eliminating any need of certification authorities for certifying identity-key associations. We allow 1am users to have human readable user names, which may collide. We resolve user name collisions by comparing public keys that are extremely unlikely to collide. The UI is designed to make the implications clear to users when the same user name is associated with different keys.

Message Authenticity: Like all public communication systems, 1am is subject to rumors and misinformation. Conventional solutions relying on centralized filtering or censorship cannot be used, due to 1am's decentralized architecture. To increase the cost of disinformation campaigns, we protect the authenticity of 1am posts with digital signatures, which are individually verified by each receiver. The intention is for credible 1am users to gradually build up positive reputations. In addition, it would be possible to use distributed spam or propaganda detection techniques [96]. There is no requirement for users to associate their actual identities with their 1am identities.

Multimedia Support: Bandwidth is a scarce resource in device-to-device networks: 1am cannot support content of arbitrary size. Our study shows that it is feasible to support images. However, without re-designs of the message delivery protocol or major improvements on wireless transceiver energy efficiency, video sharing cannot be supported in our system (see Section 5.5).

Figure 4.1 illustrates the architecture of 1am. From the bottom up, the first layer is in charge of **contact discovery and link establishment**. It detects contact opportunities, establishes an 802.11 ad hoc channel with discovered peers, and informs higher level protocols. Note that this layer is not 1am-specific and should be shared by all DTN applications. The second layer is the **message delivery protocol** that routes 1am messages to intended receivers. Finally, when a 1am message arrives, the **UI** is informed and the message is stored in a **database** for future use, e.g., re-transmission. The UI's displays messages according to given filtering rules, e.g., those from immediate wireless communication neighbors or "followed" identities.

We choose 802.11 ad hoc technology over Bluetooth for device-to-device communication, because the former has longer transmission range. We avoided WiFi Direct due to the inconvenience of its pairing process, which requires human involvement, as well as its lack of broadcast support.

Although modern smartphone hardware and operating systems support 802.11 ad hoc

communications, at the time of the study an API was not provided by Android. During our 1am deployment study, to avoid requiring users to "root" their phones, we used an ad hoc network emulation layer to identify direct wireless connectivity, deliver messages, and gather data[1] (See Section 4.3.1 for more details).

4.2.2 Message Delivery Protocol

We use a flooding protocol for message delivery for two reasons. First, flooding protocols can be used to support arbitrary filtering rules. At this stage we are not sure which rules are most appropriate for DTN-based microblogging services and therefore avoid any premature optimization tying the system design to specific rules. Second, in non-congested situations, e.g., for applications with small traffic volume, flooding protocols have the best message delivery performance and worst energy consumption among all routing protocols. Therefore, they serve as good reference points for understanding the limitations of the underlying network and provide guidance for further optimization.

To avoid redundant message transmissions, we exchange Bloom filters during each contact to learn about each node's received and not-yet-received messages. Bloom filters are compact data structures for storing elements and testing set membership [97]. They are widely used in DTN routing protocols for their space- and bandwidth-efficiency. Specifically, each node stores all its received messages in a Bloom filter that is exchanged at each contact. The other party could then test which of its own messages have not been received by the node. Classic Bloom filters use 14.4 bits per key for a 10^{-3} set membership test error rate [97]. This is much smaller than the original message it documents.

4.3 1am Dataset Description

This section describes the data collection platform of the 1am deployment, and provides basic statistics on the gathered traces. It gives an overview of participants' motion and contact patterns, and provides background for understanding the performance numbers in Chapter V.

4.3.1 Deployment Platform

Commodity Android devices do not support appropriate 802.11 ad hoc communication [98]. Enabling it would require rooting participants' phones, which we consider an

[1] We subsequently developed an application-level method that supports direct device-to-device WiFi communication on Android platforms.

unacceptable barrier to study participation. Instead, we applied a widely used technique in the DTN research community to estimate 802.11 contact opportunities. To simplify, two devices with simultaneous visibility of common WiFi access points are considered to be capable of direct communication [95, 41].

We use a data collection platform we designed for assisting DTN protocol deployments, called MANES [99]. It gathers time-varying location and wireless environment measurements. It also estimates contact opportunities based on WiFi scan results and provides applications with an emulated 802.11 ad hoc communication interface. MANES replaces the "contact discovery and link establishment" layer in Figure 4.1.

MANES has two components, an Android client that gathers GPS and WiFi scan measurements, and a backend server that aggregates and stores user traces. Location data are uploaded as soon as they become available to support up-to-date topology estimation, as the server uses the estimated topology to relay packets between 802.11 neighbors.

To save energy for the client, automated sampling frequency adjustment is used when taking both WiFi and GPS measurements. If there are no significant changes since the last measurement, further WiFi scans are postponed for 120 seconds and further GPS scans are postponed for five minutes. Otherwise, the WiFi sampling frequency is 30 seconds and the GPS sampling frequency is one minute. Because humans are usually stationary, the average WiFi measurement period in the collected traces is 114.5 seconds.

4.3.2 Participant Recruitment

1am traces are collected from fully informed users with their consent. We released 1am to the university community and advertised via posters, gifts of promotional water bottle, and word of mouth. We had 291 users, each of whom used the application at least three days from March to October in 2013. Most were university students. The median trace duration per user is 24 days. We use the first 31 days of traces, the month with the maximum number of participants, for the analysis in the rest of this chapter. We refer to this period as the *study period*. During the study period, we had 111 active study participants, with an average of 85 active participants each day.

4.3.3 Contact Statistics

During the study period, 98,072 inter-participant wireless communication contacts were made, with an average of 20 contacts, six of which were unique, per user, per day. 10% of users made no contacts. The average contact rate is 0.0329 contacts per pair per hour, which is much higher than the rate of other WiFi-based contact traces widely used in the

Table 4.1: Comparison of Contact Traces

	U-M	Dartmouth	1am	UCSD
Year	2015	2003/2004	2013	2002
Device	Laptop/Smartphone	Laptop/PDA	Smartphone	PDA
Duration (days)	30	114	31	77
Granularity (seconds)	NA	300	114.5	120
Devices participating	119,055	6,648	111	273
Number of contacts	523,951,695	4,058,284	98,072	195,364
# contacts/pair/hour	0.00025	0.000067	0.033	0.0028

DTN community (see the last row in Table 4.1). We suspect there have been changes in normal smartphone users' behaviors since those traces were gathered. In our study, users were carrying their own modern, compact, highly functional smartphones and in the UCSD [46] and Dartmouth traces [45], they were carrying either limited-functionality PDAs or laptops; these devices were carried less frequently than modern smartphones, resulting in decreased contact frequency. It is also possible that 1am users tend to recruit friends and acquaintances with whom they have frequent contact. Previous research shows correlation between physical proximity and social relationships [100].

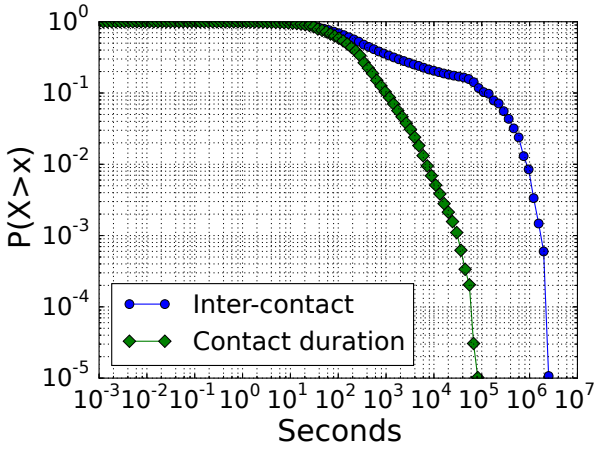


Figure 4.2: Contact duration and inter-contact time distributions, each following (truncated) power law distributions.

Figure 4.2 shows the distributions of contact duration and inter-contact time. The contact duration has a median of 149 seconds. The inter-contact time has a (truncated) power law distribution in the 30 s–1 day range, with a coefficient of 0.3. This is consistent with prior research findings [41].

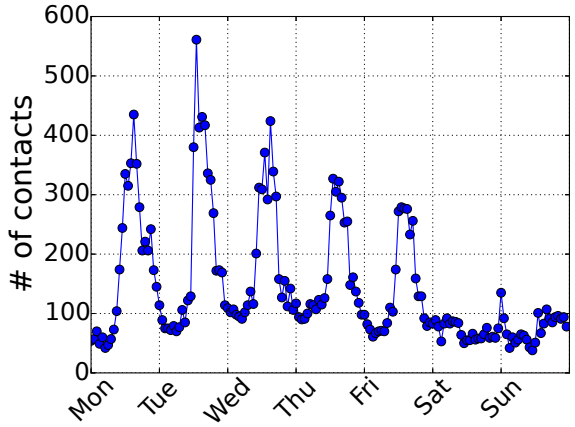


Figure 4.3: Average # of contacts by weekday, showing a strong weekday/weekend pattern.

4.3.4 Diurnal Patterns

Figure 4.3 and Figure 4.4 illustrate a weekly and diurnal pattern of contact numbers, which is expected for a population that is working or studying during weekdays. Note that a pair can meet and have contacts multiple times in a day. On an average weekday (Figure 4.4, "Weekdays"), the contact number starts to rapidly increase at 9 a.m. and is at its highest from 1 p.m. to 3 p.m. It then gradually decreases, reaching a minimum at around 3 a.m. On an average weekend (Figure 4.4, "Weekends") the contact number stays almost constant at the minimum level of the average weekday. This difference would be expected for a population in which most contacts result from co-location during weekday work or study.

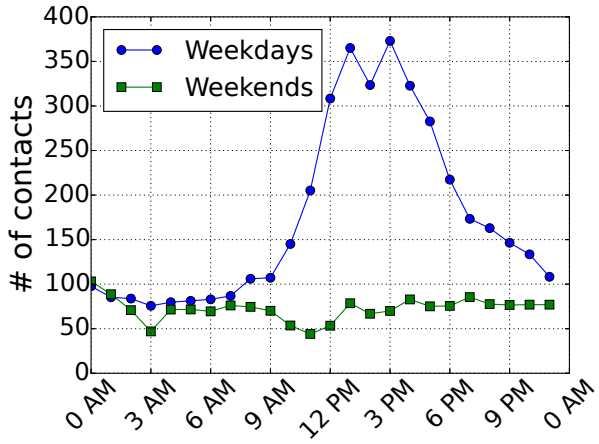


Figure 4.4: Average # of contacts by hours, showing a strong diurnal pattern with its peak in the early afternoon.

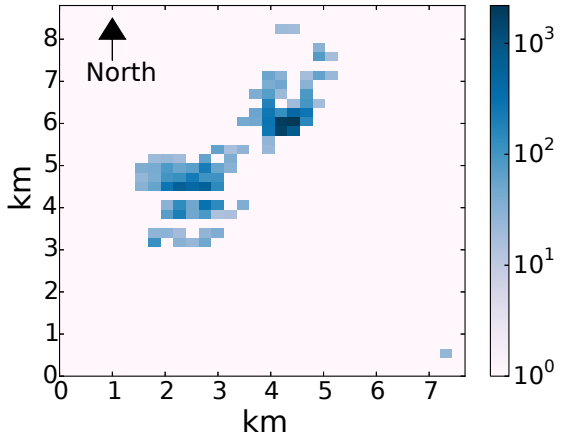


Figure 4.5: Geographic distribution of the contacts, illustrating the two university campuses. The majority of contacts occured in the Northeast Campus.

4.3.5 Geographic Patterns

To better understand the study population, we examined the geographic locations of contacts. Not all contacts are associated with GPS readings (only 27.6% are), but all are associated with WiFi scan results. This is expected, as humans typically spend most of their time indoors and GPS readings are often not available indoors. We used the following algorithm to estimate GPS locations based on WiFi scan results. We first estimate the GPS locations of WiFi access points using location samples containing both GPS readings and WiFi scan results; the location of one access point is the average of the GPS locations for all measurements with WiFi scans containing that access point. We then estimate the location of each contact event associated with only WiFi scans as the estimated location of the access point with the highest signal strength in these scans. This algorithm allows us to estimate the locations of 99.4% of the contact events.

To simplify explanation and analysis, we divide the entire area into 100 m×100 m discrete locations that we associate contact events with. There are 239 discrete locations with at least one contact, each. Contacts are distributed unevenly. If the locations are ordered by frequency of contacts, we find that 90% of the contacts occur in the first 25% of locations. Figure 4.5 shows the map of the first 100 locations, with the shade indicating number of contacts. These locations are visually clustered into two components, clearly outlining the two campuses of the university that where the study was done. The Northeast Campus, where the study was most heavily promoted, has more contacts.

To further understand the context of the contacts on each campus, we consider their temporal patterns. Figure 4.6 shows the average number of contacts in the two campuses by

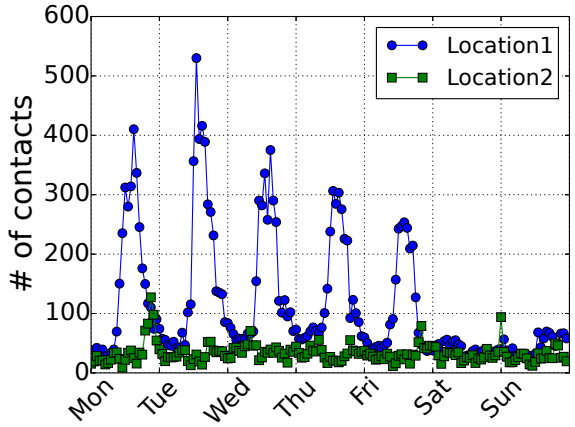


Figure 4.6: Total # of contacts in the two campuses in an average week. "Location1" (the Northeast Campus) shows weekly and diurnal patterns. "Location2" (the Southwest Campus) does not have obvious patterns.

weekday. The Northeast Campus curve ("Location1") has clear weekly and diurnal patterns, while the Southwest Campus curve ("Location2") does not. This suggests that most 1am users work in the Northeast Campus (in fact in a few very concentrated areas of that campus, i.e., the darkest few grid elements), instead of the Southwest Campus.

4.4 U-M Dataset Description

This section describes the collection and basic statistics of the U-M dataset, the contact traces of all WiFi-enabled devices on the University of Michigan Ann Arbor campus, who have ever logged into the campus WiFi network during our study period.

The U-M dataset is collected through passive collection of WiFi association histories. Nearly all buildings on this campus, including residence halls, are covered by a university managed WiFi network. With the help of U-M Information and Technology Services, we obtained all the association/disassociation activities on this network in the month of April 2015, which involves 6,277 WiFi access points and 119,055 client devices. The recorded device number is nearly twice the U-M population (69,408[101, 102]), likely because each person owns more than one devices. During this month, each device has on average 13 days with recorded WiFi activities, and an accumulated daily association duration of 3.66 hours on these days.

Figure 4.8 shows the number of recorded devices per hour in an average week, clearly reflecting the diurnal and weekday/weekend patterns of campus activities. There are as many as 35,000 devices in the busy hours of weekday early afternoons. Even in the most

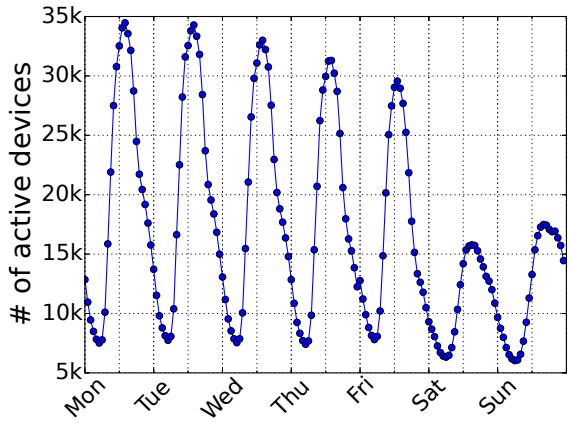


Figure 4.7: # of recorded devices per hour on the U-M WiFi network in an average week.

quiet hours of weekend nights, there are more than 5,000 devices.

The remaining part of this section describes the statistics of the U-M contact trace dataset, laying a foundation for understanding the network behaviors in Chapter V.

4.4.1 Contact Statistics

Table 4.1 summarizes the U-M contact dataset. During the course of a month, we recorded 524 million contacts among 119,055 devices. The average number of contacts per pair per hour is 0.00025. We also listed the aforementioned Dartmouth contact trace in comparison, which was collected more than a decade ago [95]. The U-M trace has around 20X increase in device numbers and 100X increase in contact numbers, resulting in 4X increase in average contact rate. This is likely a result of increased mobile device adoption rate and WiFi coverage in the last decade.

The forth column in Table 4.1 lists a much smaller scale dataset, the 1am dataset, that we collected through active participant recruiting on the same location, the U-M campus. As shown in the table, this dataset has a much higher average contact rate than the U-M one, likely because this subgroup were sampled in a biased way—we recruited most of them from one campus location on one workday. Interestingly, we calculated the average contact rate between pairs with at least one contact in the U-M dataset, and the resulting contact rate is almost the same (0.032) as that of 1am. This result indicates that instead of having a uniform contact rate with all others, people only get in contact with a small group of people, regardless of how large the population size is. We will frequently refer to and compare with the 1am traces in the remaining parts of this chapter, as contrasting these two datasets helps deepen our understanding of the properties of human contact networks.

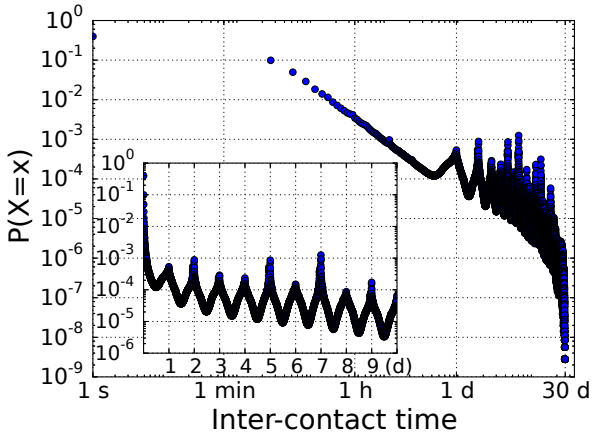


Figure 4.8: Inter-contact time distribution for the U-M dataset.

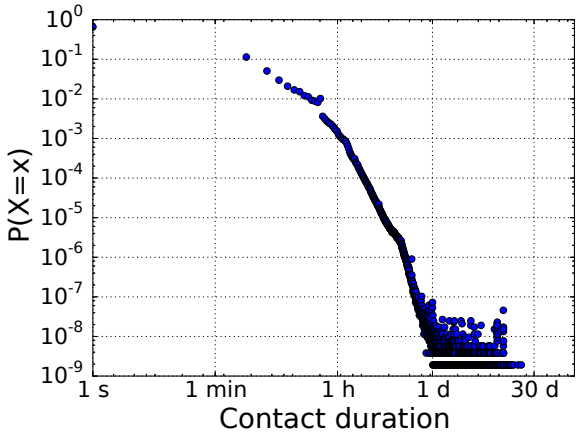


Figure 4.9: Contact duration distribution for the U-M dataset.

Figure 4.8 shows the inter-contact time distribution of the U-M dataset in log-log scale. It is a truncated power-law distribution modulated by a periodic trend. The median inter-contact time is 8.97 minutes. The inset plot with a linear X-axis (in unit of days) illustrates the periodicity more clearly, showing continuous peaks at 24, 48, 72 hours... This reflects the well-known phenomenon that people's movements are highly regular and they likely return to same locations day by day [58]. There are multiple peaks, instead of only one at exactly 24 hours, because people do not repeat their lives exactly every day. After a contact is missed because one person did not follow his/her routine that day, the pair have to wait for multiple days until both show up in their routine "rendezvous" again. Note that the inter-contact time of 7 days has higher probability than that of 1 day, likely because weekly patterns are more dominant in campus activities, e.g., going to a class once a week. Figure 4.9 shows the contact duration distribution. It is also a power-law distribution with a

Table 4.2: Comparison of Network Graphs

	U-M network	Facebook [103]	Twitter [104]	Internet [105]
Year	2015	2011	2007	1999
Nodes	119,055	721,000,000	87,897	6,374
Links	168,000,000	68,700,000,000	829,247	NA
Scale-free?	No	No	Yes	Yes
Avg. degree	2,863	190	18.9	3.82
Avg. path length	2.3	4.7	4.12 [2]	3.72
Assortativity coeff.	0.357	0.226	0.58	-0.189 [3]
WCC size	98.64%	99.91%	93.03%	NA
Cluster coeff.	0.473	[0.11,0.32] [4]	0.106	0.24

steeper slope. The median contact duration is 76 seconds.

4.5 Static Human Contact Graph Analysis

Graph analysis is a powerful tool to study complex networks. To gain insights on the structure of human contact networks, we construct a static graph out of the U-M contact traces and examine its basic properties. The contact graph is constructed in the following way. A vertex represents a mobile device. An edge is drawn between a pair of vertices as long as there is at least one contact between the two devices during the one month study period. The resulting graph includes 119,055 vertices and around 168 million edges. Table 4.2 summarizes the basic properties of the U-M human contact graph. We also listed a few well-known networks for comparison: the friendship graph of Facebook users [103], the following relationship graph of Twitter users [104], and the connectivity graph of Internet autonomous systems [105].

Similar to the two social networks, the human contact network has positive assortativity coefficient and a giant connected component containing the majority of the vertices. On the other hand, three properties of the human contact network are noticeably different. First, the average node degree is much larger. An average device encounters nearly 3,000 others on campus, while an average Facebook or Twitter user has less than 200 friends; we encounter far more people than we keep friendship with. Second, the average path length is shorter. An average path length of 2.3 means that it only takes 1.3 people to connect any two people, suggesting that the campus community is a tightly connected "small world". Third, the cluster coefficient is much higher, even higher than that of the Facebook friendship graph.

[2]Not reported in [104]. Extracted from another Twitter dataset [106].
[3]Not reported in [105]. Extracted from another Internet AS graph dataset[107].
[4]Not directly reported in [103]. We calculated this range according to Figure 1(b) and Figure 4(a) [103].

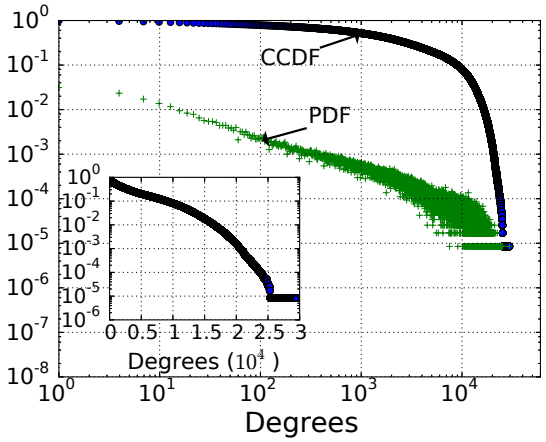


Figure 4.10: The degrees distribution.

This is likely because all people in one location are probably in contact with each other simultaneously.

These basic properties outline a densely connected, locally clustered, small-world network. We now take a more closer look at its structure, in particular the degree distribution and assortativity.

4.5.1 Degree Distribution

Figure 4.10 shows both the probability density function (PDF) and the complementary cumulative distribution function (CCDF) of degrees in a log-log plot. The distribution is skewed to the right; the median degree is 1120 and the average is 2863. We make two observations regarding the shape of the distribution. First, the distribution has a power-law shape in the left end (below the degree of 1,000), as both the PDF and CCDF are linear in this region. Second, the distribution has an exponential tail in the right end (beyond the degree of around 17,000), as the log-linear inset plot shows a clearly linear curve in that region. It is not clear how to characterize the region between 1,000 and 17,000. Nevertheless, it is safe to conclude that the degree distribution of our human contact network does not follow power-laws, whose CCDFs should be straight lines on a log-log plot. This sets apart the human contact network from most social networks and the Internet, since they are shown to have power-law degree distributions [104, 105, 108]. On the other hand, due to the trend on the lower end the degree distribution cannot be classified as exponential, either, which sets apart the human contact network with random networks that have uniform mixing patterns [109].

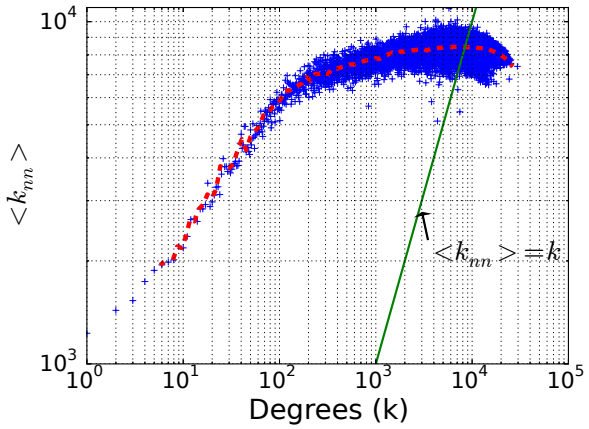


Figure 4.11: The scatter plot of average neighbor degrees ($<k_{nn}>$) versus node degrees.

4.5.2 Assortativity

We also examine the mixing patterns of the human contact network. Figure 4.11 illustrates the scatter plot of the average degree of a node's neighbors ($<k_{nn}>$) versus the degree of the node itself (k), showing a clearly positive correlation. The assortativity coefficient is 0.357. To show the trend more clearly, we plot the sliding window average in the dotted line. As illustrated, $<k_{nn}>$ increases almost monotonically with k until around k=10,000, also the point where $<k_{nn}>$ turns smaller than k. Beyond this point it is too hard to find peers with same degrees.

Assortative mixing is not surprising in a campus setting, as people are usually divided by academic programs and we tend to mostly associate with others in the same programs and of the same years. For example, freshman students sitting in the same "College Physics" class likely also take the same "College Chemistry" class. Likewise, PhD students who attend the same class probably also go to the same academic building for research.

This result confirms the resemblance between human contacts networks and social networks, as most social networks also have positive assortativity coefficients [104, 103, 107]. It is not surprising, since location proximity often indicates social proximity, and vice versa [100]. On the other hand, the Internet is shown to have a negative assortativity coefficient [107], which reflects its hierarchical structure that edges mostly exist between nodes of different levels (nodes with higher levels have larger degrees). This contrast confirms that the structure of our human contact network is not hierarchical.

To summarize, the human contact network on the U-M campus bears more resemblance to social networks than to the Internet. Compared to online social networks, the human contact network has a less heterogeneous degree distribution. It is also more densely

connected and locally clustered, as shown in Table 4.2.

4.6 Summary

According to statistics in Section 4.3, the 111 users in the 1am traces are a small community of people whose contacts are generally a result of work- or study-related co-location. Most of their contacts occur within a few work/study areas. Their motion patterns are less concentrated and correlated during nights and weekends. They continue to have contacts at these times, but with lower frequency.

On the other hand, the U-M dataset involves all (119,055) WiFi devices on the U-M campus, likely covering a significant portion of the campus population. The 1am dataset is essentially a subset of the U-M dataset, about 0.1% of the total size. The two datasets show similar diurnal patterns. Their inter-contact time and contact duration distributions also have similar shapes, with the larger dataset showing longer inter-contact time and shorter contact duration. The average contact rate, i.e., number of contacts per pair per hour, of the larger dataset is about 100X less than the smaller dataset. However, if we only consider pairs with at least one contact in the larger dataset, the resulting contact rate is almost the same (0.032) as that of the 1am dataset. This indicates that the 1am dataset is a biased, instead of random, sample of the U-M dataset. As the 1am dataset are collected from volunteering users of our 1am microblogging application, this dataset likely represents initial user groups of DTN applications.

CHAPTER V

Performance and Energy Consumption Analysis of a Large Scale Delay Tolerant Network for Censorship-Resistant Communication

5.1 Introduction

Delay Tolerant Networks (DTNs) composed of human carried devices have the potential to support blocking-, censorship-, and surveillance-resistant communication applications. Unlike the Internet, the non-hierarchical, device-to-device structure of DTNs makes it difficult to block, modify, or monitor a large portion of network traffic via attacks on a few devices.

The demand for the functionality offered by infrastructureless communication systems is suggested by the response to intentional widespread service disruption for users of social media applications such as Twitter and Facebook during the Arab Spring [6], ongoing political censorship in Iran and China, and growing awareness of global surveillance of individuals without reasonable cause to believe they have committed crimes. Recently in Hong Kong, in response to the threat of potential denial of service attacks on the Internet-based Instagram, there were 200,000 downloads of a local mesh networking chat program called FireChat between 28 and 30 September 2014 [34]. This application supports text messaging and photo transfer mediated by device-to-device Bluetooth connections [34]. Popular social networking/microblogging services are now commonly censored throughout much of the world. For example, Twitter is coerced to censor its users on a country-by-country basis by the threat of punishment by various governments [110]. In response to this trend, we have evaluated the feasibility, performance, and energy consumption of DTN systems composed of commodity smartphones and tablets when subjected to random and targeted denial-of-service and censorship attacks.

DTNs composed of commodity smartphones and tablets are influenced by constraints on battery energy, computation speed, and network throughput. We evaluate the potential of DTNs to support local communication using a flooding protocol variant that avoids redundant transmissions. Message delivery rates and latencies, as well as battery energy overheads, are evaluated using two datasets of wireless device-to-device connectivity traces. The 1am dataset involves 111 users on the University of Michigan campus for the month of March 2013, and was gathered via an active deployment of a prototype microblogging application that we developed. The U-M dataset records all 119,055 WiFi devices on the same campus for the month of April 2015, and was gathered via passive collection of Wifi association events on the campus WiFi network (See Chapter IV for details).

In summary, we make the following contributions in this chapter.

- Performance: Our analysis indicates that in a college town in which 0.2% of the population install a DTN app, the system has a median delivery rate of 0.85 after 72 hours and a median delivery delay of 13 hours. When all 119,055 mobile devices are involved, the campus-wide DTN achieves a message delivery rate of 0.71 after 24 hours, and 0.95 after 72 hours. The median delivery delay is 10.9 hours. We pay special attention to the system's performance variations and study their causes.

- Utility: We found that the delivery delays of DTN flooding protocols follow a power law distribution that varies from hours to days. We compare the distribution of the network's message delivery delays with the distributions of user response delays of various popular online applications, and show that this campus-wide DTN is appropriate for sharing non-time sensitive information such as Flickr photos and weblog articles.

- Robustness to denial-of-service and censorship attacks: We found that attacks that randomly remove 90% of the network participants only reduce delivery rates to the remaining participants by less than 10%. Targeted attacks, which remove nodes that have most contacts, were more harmful, causing the delivery rate to decrease precipitously when half of the participants were removed. However, even when subjected to targeted attacks, the network only suffered a less than 10% decrease in delivery rate when 40% of its participants were removed.

- Energy consumption: We analyzed the impact of participating in large scale DTNs (119,055 nodes) on smartphone battery lifespan. Based on a measurement-based wireless communication aware smartphone power consumption model, our analysis

shows that supporting text messages using a naive epidemic flooding protocol consumes 87.5% of a typical smartphone battery in 14 hours. We show that an energy efficient variant of the epidemic flooding protocol has around 10X decrease in energy consumption, while keeping comparable message delivery performance. Supporting text messages using this protocol consumes 13.7% of a typical smartphone battery in 14 hours.

The remaining parts of this chapter are organized as follows. Section 5.3 reports the measured message delivery performance of the two datasets. In Section 5.4 we study the robustness of DTNs against random and targeted node failures. Section 5.5 develops a measurement-based energy model for supporting DTN applications on commodity mobile devices and reports energy overheads for the two datasets. Finally, Section 5.6 concludes this chapter.

5.2 Related Work

The idea of using infrastructureless networks to support censorship-resistant communication is shared by various existing work [111, 43, 27]. Al-Akkad et al. use mobile ad hoc networks (MANETs) to enable continuous tweet dissemination when network infrastructure is disrupted [111], which has fewer applications than DTN-based systems because MANETs require instantaneous end-to-end connections. Hossmann et al. describe a DTN-based Twitter application for disaster communication. They do not provide evaluation results due to the lack of contact traces specific to disaster scenarios [43]. Our goal is providing daily information exchange services in local communities. We therefore use daily human contact traces to study achievable message delivery performance and energy consumption in such systems.

Due to the difficulty of collecting large-scale human contact traces, most existing work on DTNs [33, 95] uses a few existing traces [45, 46] for evaluation. The Dartmouth traces record the association histories of all participants on Dartmouth College's campus WiFi network [45]. Most of the recorded devices are laptops, whose mobility and contact patterns may differ from smartphones. The UCSD traces were collected around a decade ago from participants who used experimental PDAs during the deployment [46]. Their traces show a trend of declining user engagements [46] that is uncommon for normal smartphone users. Our analysis used two sets of traces we gathered in 2013 and 2015, respectively, through active installation of measurement software and passive collection of WiFi association events. They are representative of the mobility and contact patterns of modern smartphone and laptop users.

We are unaware of existing measurement-based models for the energy consumption of common DTN routing protocols or applications. Socievole et al. evaluated the energy consumption of five DTN routing protocols via simulation [112]. They modeled Bluetooth communication, while we modeled 802.11 ad hoc communication. In addition, their analysis considered networks with a maximum size of 200 nodes, while we considered networks with 119,055 nodes.

Various existing work on improving the energy overhead of DTN flooding protocols assumes a simple linear relationship between the network's energy consumption and the average number of copies per message [113, 114], not considering the cost of learning each node's received and not-yet-received messages to avoid redundant transmissions. Our results show that this cost is a significant contributor of energy overhead, especially in large networks.

5.3 Message Delivery Performance

In this section we evaluate DTN system's message delivery performance using two metrics, **message delivery rate**, the percentage of intended receivers that have successfully received a message, and **message delivery delay**, the elapsed time from message initiation to its successful delivery.

5.3.1 Methodology

We use epidemic flooding to deliver messages in the network. Specifically, every node that has received the message transmits it to all its contacts that have not received the message yet. This produces the shortest path in time between a sender to any other receiver node on the network. Evaluating the network delivery performance using this protocol generates an upper performance bound. Note that epidemic flooding does not have optimal bandwidth overhead, and in congested situations some messages may be forced to follow circuitous routes, causing performance degradation. However we do not complicate the problem with bandwidth concern for now and simply assume adequate bandwidth for all messages.

Instead of implementing and simulating the epidemic flooding protocol on each node, we algorithmically calculate the shortest path message distribution tree for each initiated message, according to the contact history of involved nodes. Specifically, we implemented Dijkstra's algorithm on the temporal graph of contacts where path lengths represent elapsed time between contacts.

In order to fully explore the dynamics of the underlying network, we sample and evaluate multiple message initiations from different senders and at different times. Our following analysis is based on the aggregated shortest-path trees associated with all message initiations. For the 1am dataset, we cause every device in the trace to initiate a message every six minutes. We do not initiate messages from a particular smartphone when it is inactive, i.e., in sleep mode or powered off.

The sampling method for the U-M dataset is more complicated. It would be ideal if we could simply let every node constantly send messages. However, since the network is huge—it takes nearly 40 GB to bring all the contacts in memory—and it takes around 20 minutes to calculate one flooding tree, we cannot afford this large scale simulation. Instead, we first randomly choose a few initiation days. We then divide each initiation day, plus the day following it, into half-hour slots and in each slot, repeat the process of randomly choosing an initiator to send a message. We continuously initiate messages on these chosen days, in order to capture the daily dynamics of the network. As it turns out, the three initiation days that we randomly choose is a Tuesday, a Thursday, and a Saturday. For the reported results in this thesis, we have simulated in total 21,163 messages.

It does not make sense to initiate messages when the device is powered off, and we try not to. However, it is difficult to infer device sleep times for the U-M dataset, as all we have is their WiFi association history. We note that a sleeping or powered-off device does not associate to any APs and long period of disassociation likely indicates sleep or powered-off state. According to the association traces, the daily maximum disassociation duration from noon to noon is on average 5 hours. Therefore, we use the following method to infer sleep or powered-off states; all disassociation periods more than 5 hours are considered to be sleep or powered-off periods and the device does not initiate messages then.

5.3.2 Overall Performance

Figure 5.2 illustrates the progress of the simulated flooding processes over time for the U-M dataset. The X-axis shows the elapsed time in hours since message initiation. The Y-axis shows the delivery rate: the ratio of active nodes that have received the message. We only show the results for three days, as messages that take more than three days to deliver are considered outdated. Since we simulated thousands of flooding processes, we use percentile lines to show their performance distribution. X-percentile line means X-percent of flooding processes have smaller delivery rates than that of the line. For example, at 12 hours, following the increasing direction of the Y-axis, the slowest 10% of messages are delivered to less than 0.25 of the network, messages falling into the 10- to 20-percentile are delivered to [0.25,0.32) of the network, . . . , and messages in the 70- to 95-percentile are delivered to

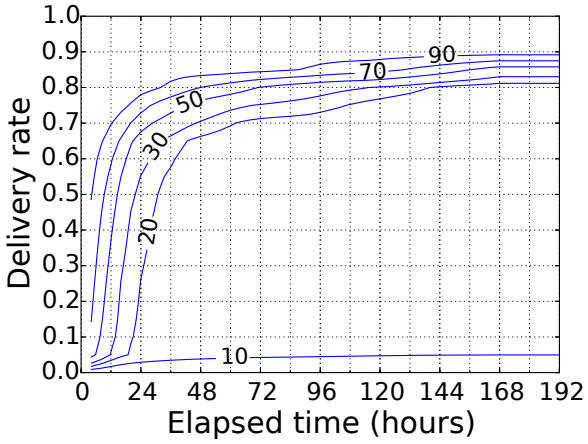


Figure 5.1: 1am's message delivery rate over time, showing two stages with significantly different propagation speeds. A point on the X-percentile line represents the X-percentile delivery rate at the considered time of the flooding processes.

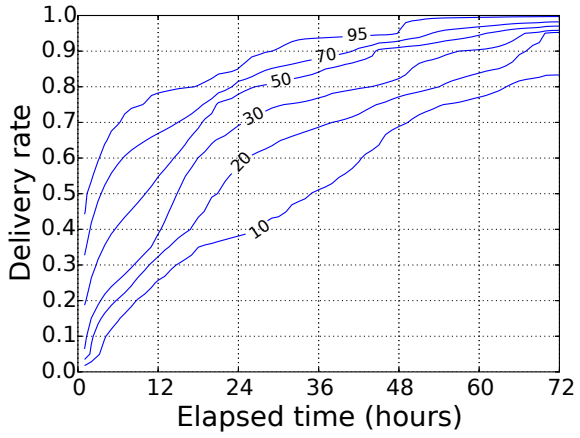


Figure 5.2: The message delivery performance over time for the U-M dataset. X-axis is elapsed time in hours. Y-axis is the delivery rate. A X-percentile line means X-percent of flooding processes have smaller delivery rates than that of the line.

[0.68, 0.89) of the network. The 50-percentile line shows the median delivery rate of all the processes. For example, at 12 hours, the median delivery rate is 0.55.

According to our simulation, for the U-M dataset, 24 hours after message initiation, the average deliver rate is 0.71 and the median is 0.79. The slowest 10% are delivered to less than 0.4 of their receivers, and quickest 5% are delivered to more than 0.85 of their receivers. After 72 hours, the average deliver rate is 0.96 and the median is 0.99. Even the slowest 10% reaches 0.83 of their receivers.

Similarly, Figure 5.1 shows the message delivery performance over time for the 1am dataset. At 24 h, the median delivery rate is 0.68. The 50th percentile line shows that it takes a median of 14 hours to reach 60% of participants, 24 hours to reach 70%, and a week to reach 85%. The fastest-spreading 10% of messages (the 90th percentile line) take 6 hours to reach 60% of participants, 12 hours to reach 70%, and a week to reach 90%. The slowest 10% of messages (the 10th percentile line) reached almost no other participants, because 10% of participants had no contacts during the study period.

Figure 5.2 and Figure 5.1 show surprising consistency in terms of message delivery performance. Recall that the 1am dataset only includes 111 participants, essentially a small subgroup containing less than 0.1% of the U-M dataset participants. Despite this huge difference in network size, the median cases of the two datasets almost have the same performance. The delivery rates of 1am's median case at 12, 24, and 72 hours are 0.5, 0.7, and 0.8, while the numbers for the U-M dataset are 0.55, 0.79, 0.99. The difference is even smaller if we only consider nodes on the largest connected component as intended receivers; the largest connected component of the 1am dataset occupies around 90% of the network, while the number for the U-M dataset is 98.64%.

5.3.3 Performance vs. Adoption Rate

The U-M dataset corresponds to a high adoption rate of DTN applications among the campus population—recall that the total number of devices in the U-M dataset is twice the campus population. In this subsection we study the effect of varying adoption rates upon the network delivery performance.

Figure 5.3 plots the average delivery rate of the U-M dataset versus the adoption rate, which is varied from 0.2% to 100%. The X-axis is in log scale. We choose subsets according to a uniform random distribution for any given adoption rate. Figure 5.3 indicates that if the adoption pattern of the infrastructureless network is purely random, it has difficulty taking off. As shown by the plot, below a 10% adoption rate the delivery rate increases almost log-lineally; the network has to grow users like snowballs to only keep a linear increase in performance. However, we just showed that instead of producing a 0.1 delivery rate as

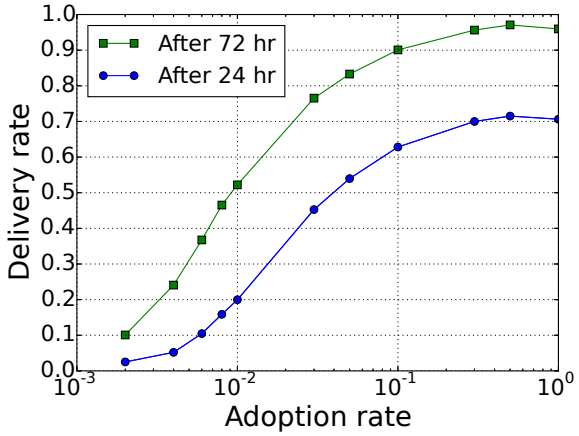


Figure 5.3: Average delivery rate as a function of adoption rate.

predicted by Figure 5.3, the 1am dataset, which only corresponds to a 0.1% adoption rate, has about the same performance as that of the 1 adoption rate case. This is because 1am participants are not randomly sampled and have more co-location opportunities than random people on the U-M campus (Section 4.4.1). This observation indicates that we are likely to have near-optimal performance even in the beginning, if the initial group of users are physically close to each other.

5.3.4 Performance vs. Initiation Time

We also look into the influence of message initiation time on delivery performance for the 1am dataset. There is a significant difference between weekday and weekend initiations. Weekday initiations have a significantly faster delivery; it takes a median of 12 hours to reach 60% of the network on weekdays, compared to 30 hours on weekends. This is consistent with Figure 4.4 showing that there are significantly more contacts on weekdays than weekends.

There are also significant performance variations by hour-of-day, as illustrated in Figure 5.4. The X-axis shows the initiation hour. The Y-axis shows the delivery delay to reach 60% of the network. For example, in the median case (the 50th percentile line), a message sent at 11 a.m. only takes 10 hours to reach 60% of the network, while another sent at 8 p.m. takes 21 hours.

In general, the closer before noon a message is initiated, the shorter its delay. The later after noon a message is initiated, the longer the delay. The shortest delay occurs at around 12 p.m., and the longest delay occurs at around 7 p.m. This trend is consistent with 1am's daily contact dynamics. The majority of the contacts occur between 10 a.m. and 6 p.m. (Figure 4.4). Messages initiated before this period have the advantage of using all

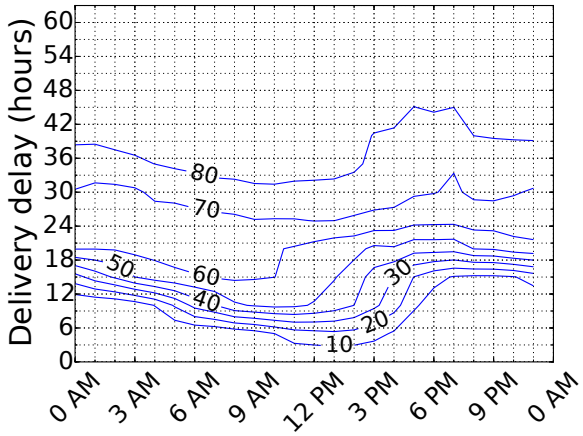


Figure 5.4: Delivery delays (to reach 60% of the network) of messages with different initiation hours, showing significant variations. A point on the X-percentile line represents the X-percentile delivery delay of the considered initiation hour.

these contact opportunities. Those initiated after the start of this period have fewer contact opportunities and longer delays.

5.3.5 Message Delivery Delay and Applications

Figure 5.5 plots the CCDF, i.e., $P(X > x)$, of the message delivery delays of the 1am dataset in log-log scale. The blue dots in Figure 5.6 show the CDF, i.e., $P(X \leq x)$, of the message delivery delays of the U-M dataset in log-log scale. Both distributions follow power-law. For the U-M dataset, about 22% of messages are delivered within an hour, 74% of messages are delivered within a day, and only 4% are delivered after 3 days. The median delay is 10.9 hours. Note that even though we only simulate the first 3 days of message deliver and therefore do not know the delay distribution of messages delivered after 3 days, it does not affect the CDF calculation for $X \leq 72$.

These delivery delays are in the range of hours, certainly orders of magnitude larger than the package delivery delays on the Internet, which are usually a few milliseconds. However, not all hope is lost. Users do not stare at their Facebook news feed, or read weblogs all day. They check them a couple of times per day, and should be satisfied, as long as the information arrives before they open the Facebook or weblog page.

To understand whether the performance of our network meets users' expectations, we examine the delay between the time of an item generation, e.g., a Facebook post, a Flickr photo upload, or a blog article post, and that of an access to the item. In particular, we extracted usage data for four popular applications from published papers.

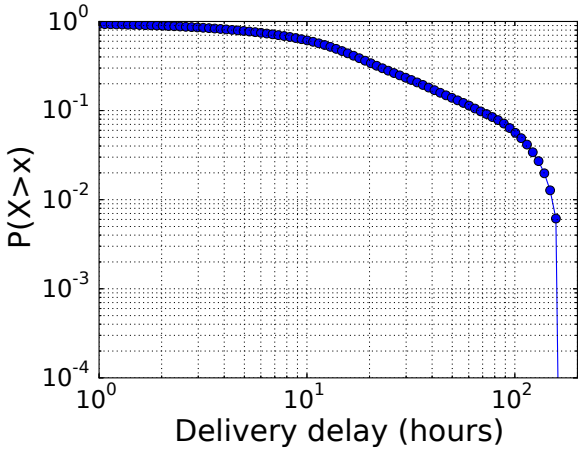


Figure 5.5: Message delivery delay distribution of the 1am dataset, which follows a (truncated) power law.

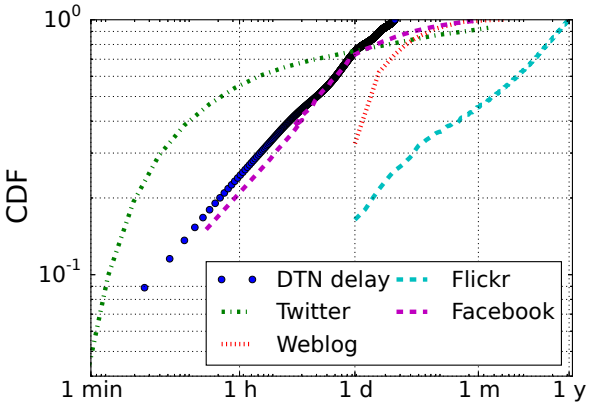


Figure 5.6: Distributions of the message delivery delays of the U-M dataset, and user response delays of various popular online applications.

- *Facebook*. An ideal metric would be the delay between when a Facebook post is shared and its reception of "likes" and comments. However, we did not find public dataset or published papers on that metric. We found that Bakshy, Rosenn, Marlow, and Adamic reported the distribution of delays between a link was shared on Facebook and a friend also shared that link [115]. Note that this is not necessarily reposting the same link and the friend may have read the link from other channels. The authors did show that the first post has influence over the friend's post, and we take this delay as an approximation as the delay between a post initiation and the audience's response time.

- *Flickr*. Flickr is a photo sharing social network website. Users may "favorite" a photo by simply clicking a button besides it. Cha, Mislove, and Gummadi reported the distribution of delays between a photo is uploaded and a user "favorites" it in their paper [116]. This is exactly the metric we are looking for.

- *Twitter*. Kwak, Lee, Park, and Moon reported the distribution of the delay between the original tweet and its retweet [106]. This is exactly the metric we are looking for.

- *Weblog*. Leskovec, McGlohon, Faloutsos, Glance, and Hurst reported the distribution of delays between a blog article is posted and other blogs link to it [117]. Note that this delay might be longer than that between the blog article is posted and a reader reads it.

We plotted these four distributions in Figure 5.6, to compare them with the message delivery delay distribution of the U-M pocket switched network. The units for the delays in Flickr and weblogs are in days, so their curves start with 1 d.

As shown in Figure 5.6, more than 50% of retweets are posted within an hour of the original tweet post, while our network could only deliver 22% of messages within an hour. It is therefore evident that our network cannot serve Twitter. On the other hand, most Flickr favorites (84%) and Weblog links (69%) are created after one day, while our network could deliver most messages (74%) within one day. Our network could therefore satisfy most Flickr photo views and blog visits. Lastly, the distributions of Facebook repost delays and our message deliver delays follow almost the same distribution, especially for regions below a day. If those two distributions are independent, we have a 50% chance to successfully serve Facebook users. Note this probability might become larger than 0.5 if the frequency that a user checks a friend's Facebook updates declines with their distance, since delivery delays on our network increases with distance.

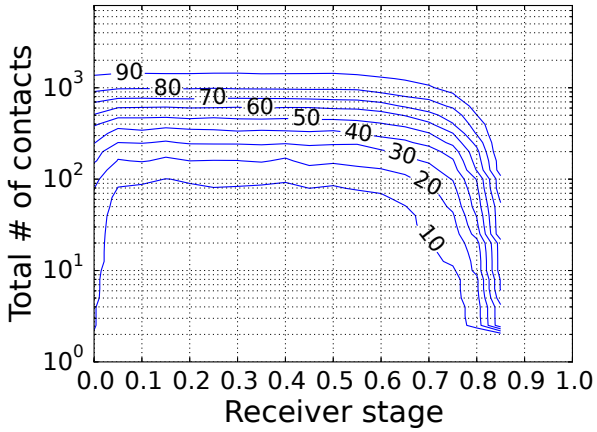


Figure 5.7: Contact number distributions of different-stage receivers in the 1am dataset, showing a transition at 0.6 delivery rate. A point on the X-percentile line represents the X-percentile contact number of receivers reached at the considered stage.

5.3.6 Delivery Speed

As shown in Figure 5.1, the message propagation speed (the slope of the percentile lines) of the 1am dataset clearly show two stages. To reach the first 60% of participants it takes 3 hours for every 10% of progress, but it becomes increasingly difficult to reach the rest of the participants. For example, the median case (the 50th percentile line) takes only 18 hours to progress from 0% to 60%, but 50 hours to progress from 60% to 80%. The power-law distribution of delivery delays in Figure 5.5 also illustrates this phenomenon.

To explain this change in propagation speed and understand why it takes so long to reach the last 40% of participants, we examine the difference in contact numbers for receivers reached in different stages of the flooding processes. Different flooding processes have different propagation speeds. Therefore, we did not use absolute elapsed time to quantify a receiver's stage. For example, 9 h is late in a flooding process that finishes in 10 hours, but is early in another one that finishes in 100 hours. To define a metric independent of propagation speed, we order the receivers in each flooding process by their reception times and set the stage of a receiver based on its comparative order, i.e., the delivery rate upon that receiver's message reception. We aggregate same-stage receivers of all the flooding processes and show their contact number distributions in Figure 5.7. The X-axis shows the stages of receivers using delivery rates. The Y-axis is a node's total number of contacts during the study period. As there are multiple nodes at the same stage, we use percentile lines to show their contact number distributions.

There is a clear change in trend at the stage associated with 60% delivery rate. Below that, the contact number distributions are almost constant. Above that, the number of

contacts per node decreases dramatically. This is consistent with Figure 5.1, which shows a constant propagation speed before reaching 60% delivery rate, and a decreasing speed after that. This result suggests that diminishing contact opportunities prolong delivery delays for the last 40% nodes.

In summary, due to the participants' differences in contact opportunities, the 1am communication system has two stages in its delivery performance. It has a constant propagation speed (3 hours per 10% progress) for the first 60% of the participants, who are well connected with each other. It is increasingly difficult to reach the remaining nodes, which have fewer and fewer contacts. Overall, with a message lifetime of one week, the system has a median delivery rate of 0.85 and a median delivery delay of 13 hours.

5.3.7 Summary on Message Delivery Performance

To shortly summarize, a community scale DTN composed of people's own mobile devices—such as the one covering the UM campus—delivers most messages within a few hours to one or two days, and it is most appropriate for sharing information that is not time sensitive, e.g., blog articles and photos. It is not good for pushing highly time sensitive news to the whole community, e.g., Twitter news.

5.4 Robustness

As a natural consequence of their distributed, non-hierarchical structures, DTNs are robust to blocking and censorship accomplished by removing or controlling network resources. Evaluating a network's robustness to resource removal can be used to determine its robustness to both attacks. The relationship between resource removal and blocking (denial-of-service) attacks is straightforward; an attacker might eliminate smartphones, perhaps via malware or by jamming their wireless signals at specific locations. The relationship with censorship is slightly more complex; censorship can be seen as a selective form of resource removal. Because censorship requires control over network devices, or their communication channels, a network structure robust to resource removal is also robust to censorship.

Resistance to attacks in which many network participants simultaneously and rapidly inject messages requires additional consideration. Non-hierarchical networks remain vulnerable to such denial-of-service attacks. However, resisting them is possible. For example, recall that 1am microblogging application where messages are signed using the private keys of the sending identities. It is practical for receiving mobile devices to detect when particular identities are sending messages at an undesirable rate and then cease forwarding them. An attacker might conceivably adopt a new identity and use it to launch an attack. That strategy

can be defeated by assigning higher priorities to messages from nodes that have previously sent messages of interest.

Increasing the cost of surveillance is a secondary goal. In many applications messages are public by design, e.g., blogs. However, there remains value in increasing the cost of reprisal-oriented network surveillance, e.g., associating messages with senders or receivers, or associating identities with individuals. The use of a non-hierarchical device-to-device network has the potential to increase the cost of reprisal-oriented surveillance because it requires a physical presence throughout a large portion of the network to observe enough traffic to associate senders and receivers. Associating identities with messages is also expensive because attackers would require a widespread physical presence, perhaps capable of wireless signal triangulation, to determine the location of the mobile device originating a particular message.

5.4.1 Attacks

We consider DTN performance when subjected to attacks which remove participating devices. These attacks prevent specific network nodes from transmitting or relaying specific (censorship) or all (blocking) messages, e.g., through malware deployment, physical attacks, or wireless signal jamming. Two specific strategies are considered. The first one, "random device removal", disables devices at random, and the second one, "targeted device removal", disables devices in order of their importance. More *important devices* have more contacts with other devices.

We also consider another class of attacks that disrupt physically local communications, e.g., via radio jamming on 802.11 frequencies [118], for the 1am dataset[1]. We call this "location removal attacks" because it disables all communication within a local area, or location. We defined a *location* to be a 100 m×100 m grid element, i.e., a region with an area similar to that covered by an 802.11b transmitter. Only "targeted location removal" is discussed because "random location removal" is less effective and we believe intelligent attackers would be capable of identifying *important locations*, i.e., ones with most contact occurrences.

We simulated the effects of both classes of attacks by recalculating the message delivery performance when subject to them. In the case of device removal attacks, all contacts involving the affected devices are removed. In the case of location removal attacks, all contacts occurring within the affected locations are removed.

[1] We did not simulate this attack for the U-M dataset as it contains no location information.

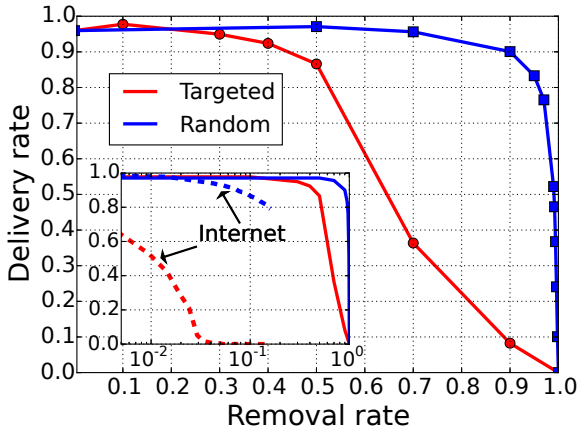


Figure 5.8: The change of average delivery rates during random and targeted node removals for the U-M dataset.

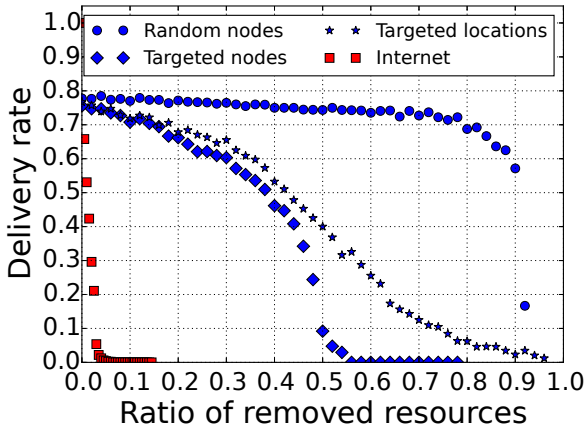


Figure 5.9: Degradation of message delivery rates under resource removal attacks for the 1am dataset. “Random node removal” has almost no effect. “Targeted node removal” is the most effective. Under “targeted node removal”, the Internet is fragmented much faster than the 1am network.

5.4.2 Attack Performance: Delivery Rate

Figure 5.8 shows the change of delivery rates during random and targeted node removals for the U-M dataset. Y-axis is the average delivery rate after 72 hours. As illustrated, random removal has almost no effect on the network delivery rate until 90% of nodes are removed. After that the network disintegrates and the delivery rate quickly falls to zero. On the other hand, targeted removal is more "effective" and it causes the network performance to degrade earlier. Under targeted removal, the network delivery rate starts to noticeably degrade after around 40% of high-degree nodes are removed. However, the degradation is much slower compared to the degradation of random removal after 0.9.

One possible explanation for this slow degradation follows. Because human contact networks are highly assortative and locally clustered, removing nodes in descending order of their degrees gradually breaks the network into smaller and smaller communities. The connectivity within each community is still preserved. On the other hand, randomly removing nodes does not affect the structure of the network, but it affects communities on all levels. Once enough nodes are removed, all communities break down at the same time.

Figure 5.9 shows how attacks affect message delivery rates for the 1am dataset, very similar with the U-M dataset. Random device removal has almost no effect on delivery rates until more than 80% of devices are removed, while targeted removal attacks more quickly impact delivery rates. Note that the targeted device removal and targeted location removal have similar impact with the first being slightly more effective.

Comparing Figure 5.8 and Figure 5.9 leads to an interesting observation that the 1am network and the U-M network have almost similar reactions to random and targeted node removals. In addition, recall that their message delivery performances are also similar. These observations suggest structural similarities between the larger human contact network and the smaller one sampled from a smaller location.

5.4.3 Attack Performance: Delay

Figure 5.10 shows the median delivery delay under random and targeted node removals for the U-M dataset. As illustrated, random removal has no effect on delivery delays until around 80% of nodes are removed. This is similar to its effect on delivery rate. On the other hand, targeted removal has immediate influence over delays. The speed of degradation increases when more than 40% of most connected nodes are removed.

The attacks have similar effects on the 1am dataset, as shown in Figure 5.11. It is interesting that contrary to its impact on delivery rate, targeted location removal has a larger impact on message delivery latency than does targeted device removal. The following

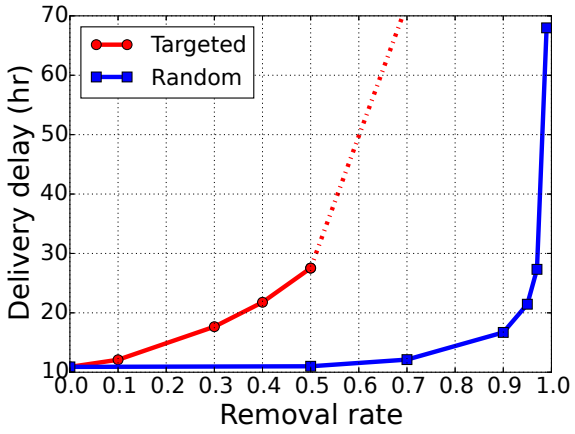


Figure 5.10: The change of median delivery delays during random and targeted node removals for the U-M dataset.

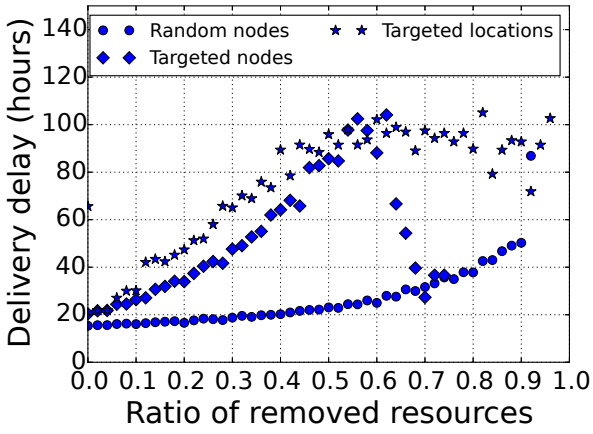


Figure 5.11: Degradation of message delivery delay under resource removal attacks for the 1am dataset. Again, "Random node removal" has almost no effect. Both targeted removal attacks are more effective, with "targeted location removal" resulting in faster delay increase.

explanation is consistent with our observations. When a location is removed, it is likely that devices will meet in other locations to complete the delivery process with higher delay. In contrast, when a node is removed, this can terminate further distribution of messages, resulting in reduced in delivery rates, without reducing the delivery delays of the remaining messages.

5.4.4 Comparison to the Internet

We are interested in comparing the performance degradation of our DTNs and the Internet when subject to resource removal attacks. Albert, Jeong, and Barabasi studied the fragmentation of the Internet when its routers are removed randomly and in descending order of edge numbers [12]. They quantity network fragmentation using the relative size of the largest cluster, which is an upper bound of the network's average delivery rate. We therefore plotted Internet fragmentation as a function of the percent of removed routers from their paper, in direct comparison with the degradation results of our DTNs.

As shown in the inset plot of Figure 5.8, compared to the Internet, the U-M network is more robust under both random and targeted node removals. For example, under targeted removal, the Internet breaks down when less than 5% of its most connected routers are removed, while the U-M network has to wait until more than 40% of its most connected participants are removed. A possible explanation is the U-M network is less heterogeneous than the Internet (recall Section 4.5.1). Note that we do not claim that the U-M network is more robust than the Internet under attacks of the same scale, because it may be more difficult to eliminate or compromise 5% of resources on the Internet than on the 1am network.

5.4.5 Summary and Discussion

Our analysis shows that it is ineffective to block or censor DTN communications by randomly removing network resources, unless the majority (more than 90%) of the resources are removed. It is more effective to target the most important resources, i.e., devices with numerous contacts and locations where many contacts occur. Even using this strategy, it is necessary to eliminate a large percentage of the 1am network, relative to a hierarchical network. The Internet is fragmented when less than 5% of its most critical routers are removed, while the 1am network does not degrade rapidly until more than 40% of its devices or locations are removed. Eliminating 40% of network devices, or radio jamming 40% of the public areas in a town or campus is likely to be an expensive undertaking for an attacker.

Table 5.1: Power Model Parameters

	Meaning	Value
N	Network size	Deployment dependent
E_f	Energy cost of send+receive one 802.11b frame	26.4 mJ
f_m	Message initiation frequency	Message type dependent (see Table 5.4)
F_m	# of 802.11b frames per message	Message type dependent (see Table 5.4)
B_{frame}	# of data bits per frame	4095×8/2
$R_{contact}$	Avg. # of contacts per pair per hour	0.00025 (U-M trace)

5.5 Energy Model

This section describes an energy model for supporting DTN applications on commodity smartphones and describes the relationship between battery depletion and network scale. We first develop an energy model for the WiFi component in 802.11 ad hoc mode based on measurements, then elaborate on the three contributors to energy consumption for DTN applications: contact discovery, communication, and computation. For the convenience of the readers, Table 5.1 lists a few parameters frequently used throughout this section.

We use a Monsoon power monitor [119] to measure power consumption. The devices used for power modeling are rooted Samsung Galaxy Nexus smartphones running Cyanogen-Mod 10.2.1-toro (Android 4.3.1). The wireless chip of these devices is Broadcom BCM4330. The CPU is a Dual-Core 1.2 GHz ARM Cortex-A9.

5.5.1 WiFi Component in 802.11 Ad hoc Mode

We measure the smartphone's power consumption for different packet transmission frequencies. The experimental setup follows. Two devices are put into 802.11b ad hoc mode and a 2 Mbps communication channel is established. One device uses the "ping" program to send periodic ICMP packets, which are padded into maximum 802.11b frames. We measure the power consumption of the other device, which is placed in sleep mode with only the WiFi component awake. The ping frequency (f_{ping}) is varied from 0.1 Hz to 100 Hz.

We observe clear changes in wireless interface power management states as f_{ping} changes. When $f_{ping} < 0.5$ Hz, the WiFi component switches on and off for each frame and consumes significant energy, as the energy cost for one power state switch is much larger than that for one send/receive operation. When $f_{ping} > 1$ Hz, the WiFi component stays in the high power state. From 0.5 Hz to 1 Hz, the frequency of power state switches gradually decreases, and

Table 5.2: Operation/State Dependent WiFi Power

P_{idle} (idle state)	210 mW
P_{high} (send/receive state)	341 mW
E_{sw} (one power state switch from low-high-low)	242 mJ
E_{ping} (send+receive one 802.11b frame (ping))	2.16 mJ
E_{send} (send one 802.11b frame (UDP))	19.6 mJ
E_{recv} (receive one 802.11b frame (UDP))	6.8 mJ

we see a decreasing power consumption despite the increase in ping frequency. Based on this result, we use a conservative two-stage model for the WiFi component. Send/receive operations separated by more than one second cause power state switches. In other words, the device stays in the high power state until there is a gap between operations longer than one second.

Table 5.2 summarizes the measured energy numbers for different power states and operations. Note that we also measured the energy cost of sending/receiving one (maximum) 802.11b frame through an application program operating UDP sockets (E_{send} and E_{recv}), as these numbers are more relevant for application energy modeling.

5.5.2 Ongoing Contact Discovery

In order to avoid relying on centralized control and scheduling, we used Zheng's and Hou's asynchronous activation algorithm [120] to check for contact opportunities periodically. The activation pattern within each period is deliberately designed so that two unsynchronized devices have overlapping activation times. The authors showed that it is possible to achieve a 9/73 duty cycle [120]. Therefore, the hourly energy cost for contact discovery is $E_{base} = P_{idle} \cdot 3,600\,\text{seconds} \cdot \frac{9}{73}$, where P_{idle} is the idle power consumption for staying awake in 802.11 ad hoc mode, the minimal power state in which devices could discover each other.

5.5.3 Communication

Communication happens upon each contact, causing the following sequence of operations. The two devices first switch into the high power state, then exchange Bloom filters to learn about each other's received and not-yet-received messages, then deliver the not-yet-received messages, and finally switch back into the low-power state. The communication energy consumption is therefore the sum of the Bloom filter exchange (we also call it metadata exchange) and message delivery energy consumptions.

5.5.3.1 Message Delivery Cost

All messages are flooded to the whole network, so the number of messages received per node per hour is the total number of messages generated by the network per hour, Nf_mF_m. f_m is the message initiation frequency (in 1/hour) and N is the number of nodes in the network. F_m is the number of 802.11b frames each message consumes (see Table 5.4 for typical message sizes). Therefore, the hourly energy cost for message delivery is $E_{msg} = E_f(Nf_mF_m)$. E_f is the energy cost of sending and receiving one maximum size 802.11b frame, $E_f = E_{send} + E_{recv}$.

5.5.3.2 Metadata Exchange Cost

The metadata of each node's received messages that are still alive are recorded in Bloom filters—recall that DTN messages have a limited lifetime, e.g., we set it to three days in the U-M simulation. We now derive the size of each node's Bloom filter. Each node receives in total $Nf_m \cdot 12\,\text{hours} \cdot 3\,\text{days} \cdot F_m$ messages per week, as we assume 12 hours of active time per day for initiating messages. Each message requires 14.4 bits in the Bloom filter (recall Section 4.2.2), so the size of each node's Bloom filter (measured in 802.11b frames) is $\frac{14.4\,\text{b}\cdot(Nf_m\cdot 12\,\text{h}\cdot 3\,\text{d}\cdot F_m)}{B_{frame}}$, where B_{frame} is the maximum number of data bits carried by one 802.11b frame.

As each node's Bloom filter is exchanged upon each contact, the energy cost per contact is $E_f\frac{14.4\,\text{b}\cdot(Nf_m\cdot 36\,h\cdot F_m)}{B_{frame}} + E_{sw}$. The extra term E_{sw} is the energy cost of switching on and off the WiFi component to enable communication (see Table 5.2). On the other hand, each node's average contacts per hour is $NR_{contact}$, as $R_{contact}$ is the average number of contacts per pair per hour. Therefore, each node's hourly cost for metadata exchange is

$$E_{md} = NR_{contact}\left(E_f\frac{14.4\,\text{b}\cdot(Nf_m\cdot 36\,\text{h}\cdot F_m)}{B_{frame}} + E_{sw}\right).$$

Overall, the hourly communication energy cost per node is $E_{comm} = E_{msg} + E_{md}$, which scales with N^2 as E_{md} scales with N^2 and E_{msg} scales with N.

5.5.4 Computation

The computation cost is the energy cost for handling received DTN messages at the application layer, which includes the cost of signing, verification, and all other operations, e.g., serialization/deserialization and database queries and insertions[2]. Each node's hourly

[2]In fact signing is only conducted once by the message initiator while every receiver verifies. We simplified the model by assuming that signing and verification occur the same number of times, leading to an upper

Table 5.3: Energy Cost of 1am Computations

	Execution time for 5,000 runs (ms)	Energy cost per run (mJ)
Signature (OpenSSL)	741.9 [121]	0.1
Verification (OpenSSL)	3819.7 [121]	0.6
Verification (SpongeCastle)	1,572,347	239
Others	65,776	10.0

energy cost for computation is $E_{comp} = Nf_m E_{mcomp}$, where Nf_m is the number of messages received per node per hour, and E_{mcomp} is the energy cost of processing one message. We plugged in the measurement results for 1am, the DTN microblogging application that we developed, in this model.

Table 5.3 lists the computation time and energy cost for various message handling operations[3]. For digital signature and verification, 1am uses ECDSA with nistp256 for their space-friendly public keys [122]. Unfortunately Android's default cryptography library, SpongeCastle [123], is much less energy efficient (2,000× less) than an unsupported native library, OpenSSL [124] (see Table 5.3). Because it is non-trivial to port OpenSSL to Android and measure its performance, we used a previous paper's reported computation time for the considered algorithm parameters, CPU model, and OS [121]. In summary, $E_{mcomp} = 0.1\,\text{mJ} + 0.6\,\text{mJ} + F_m \cdot 10.0\,\text{mJ}$, for signing, verification, and other operations, respectively. The cost for all operations except signing and verification is proportional to the message size, as reflected in the last term of E_{mcomp}.

Summing the cost of contact discovery, communication, and computation yields 1am's total hourly energy cost

$$E = E_{base} + E_{comm} + E_{comp}, \tag{5.1}$$

which scales in $O(N^2)$, as E_{base} is constant, E_{comp} is $O(N)$, and E_{comm} is $O(N^2)$. Note that this model assumes an uniform pairwise contact rate ($R_{contact}$) independent of the network size (N). In reality, $R_{contact}$ may decrease as N increases, offsetting the $O(N^2)$ trend, because an increasing number of nodes may never meet as the network becomes larger and more geographically distributed.

5.5.5 Energy Cost for Supporting the U-M Network

Using the energy model in Equation 5.1, we estimate the per device energy cost to support the U-M DTN with 119,055 devices. We consider three types of multimedia messages with

bound cost estimation.

[3]The energy cost is the product of the computation time and P_{CPU}, the smartphone's power level with CPU being awake, which is 760 mW according to our measurement.

Table 5.4: Message Sizes (F_m) and Initiation Frequencies (f_m)

Type	Message size	F_m (frames/msg.)	f_m (msg./hour)
Text	514 characters	1	0.16667
Image	100 KB	44	0.02536
Video clip	1 Mbps × 240 seconds	12000	0.00054

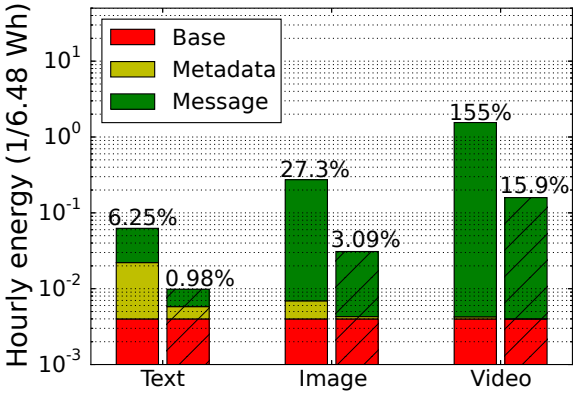


Figure 5.12: Average hourly energy consumption per device on the U-M network for text, image, and video messages.

different sizes (measured in 802.11b frames) and message initiation frequencies (f_m), as shown in Table 5.4. We consider 1am text messages that contain at most 514 characters, so one frame per text message is sufficient. The text initiation frequency is based on Twitter's average rate, around 2 per user per day [125]. We consider images of sizes typical for microblogging applications, 100 KB. The image initiation frequency is based on Facebook users' photo posting statistics [126]. We consider 352×288 video encoded at 1 Mbps, i.e., high-quality MPEG, of 4 minutes duration, the average video length on the Internet [127]. The video initiation frequency is based on YouTube's statistics [128].

The blank bars in Figure 5.12 shows the resulting average hourly energy consumption for supporting the U-M network. The Y-axis is in log scale. The unit on the Y-axis is ratio of a 6.48 Wh battery, a typical battery for mobile devices nowadays. "Base" represents the cost for contact discovery (E_{base}), "Metadata" represents the cost for running the epidemic flooding protocol, i.e., exchanging metadata on received messages upon contacts (E_{comm}), and "Message" represents the cost of handling the actual messages (E_{comp}).

As illustrated, on a DTN with around 120,000 devices, using an epidemic flooding routing protocol, supporting text messages costs a hourly depletion of 6.25% of a typical smartphone battery. Image messages cost 27.5% of the battery per hour and video messages

cost 155% of the battery per hour. Note that these numbers are far smaller than what we predicted in a previous paper, which used an hourly contact rate $R_{contact} = 0.0329$ based on the 1am dataset [57]. In fact, the average contact rate per pair on the U-M network ($R_{contact} = 0.00025$) is about 100X smaller than that of the 1am dataset, indicating that people do not meet everyone else with the same probability. This result also indicates that instead of scaling in $O(N^2)$, as predicted pessimistically by Equation 5.1, energy overhead likely scales much slower with the network size N, because the total number of contacts per device does not scale linearly with the network size.

Nevertheless, these energy overheads are non-negligible, possibly unacceptable for smartphone users, as even for the most undemanding text messages, running this application alone depletes more than half of the battery in eight hours. We observe that epidemic flooding is the most resource-consuming routing protocol and this is an upper bound on energy consumption. Since most messages only have a small number of receivers, e.g., the average number of friends that receive a Facebook post is less than 200 [103], more energy efficient protocols should be used in practice.

We consider a variant of the epidemic flooding protocol, the probabilistic flooding protocol [129, 130], for example. According to this protocol instead of relaying every message, a node decides independently with probability p whether to relay a message. As shown in Figure 5.8 (random removal), the network has almost the same performance even if only 10% of randomly chosen nodes are functioning. This indicates that a probabilistic flooding protocol with $p = 0.1$ performs as good as the epidemic flooding protocol.

This indication may not be immediately obvious as Figure 5.8 only plots the message delivery performance for the 10% functioning nodes. An informal proof follows. Consider a random node n_i who is not among the group of functioning nodes, i.e., n_i only receives messages that she is interested in but never retransmits them. Let's denote the size of the functioning group $|F|$. Now if we add n_i into the functioning group, the size of the functioning group becomes $|F| + 1$. As we have shown in Figure 5.8, the message delivery performance of these two cases with $|F|$ and $|F| + 1$ functioning nodes, should be almost identical. Therefore, in the case where n_i is functioning, its probability and delay of receiving the message follows the same distributions of reception probabilities and delays as that of the functioning nodes in the case of the size-$|F|$ functioning group. Finally, since whether and when n_i receives the message is not affected by whether it retransmits, in the real case where n_i does not function, its reception probability and delay follows the same distributions of reception probabilities and delays as that of the $|F|$ functioning nodes.

The bars filled with slashes in Figure 5.12 shows the average hourly energy consumption for the probabilistic flooding protocol with $p = 0.1$. The energy consumption is about 10

times less as each node now only handles 1/10 of the messages generated on the network. Specifically, supporting text messages costs a hourly depletion of 0.98% of a typical smartphone battery, which translates into 14% of the battery in 14 hours and is acceptable. Image messages cost 3.09% of the battery per hour, which translates into 42% of the battery in 14 hours. The video messages cost 15.9% per hour, which is not acceptable for sure. More energy efficient routing protocols are needed in the last two cases.

5.6 Conclusions

Thanks to their structure, DTNs composed of commodity smartphones have the potential to resist blocking and censorship, and make surveillance expensive. Such networks are capable of supporting text microblogging applications without much reducing participant smartphones' battery lifespan for towns with populations of 119,055. Supporting frequent image or video distribution without new dissemation ideas is impractical at similar network scales. The network was able to deliver messages to most participants within a day with only in a college town with 119,055 participating devices. It degrades more gracefully than a hierarchical infrastructure network when subjected to attacks that eliminate participants or geographic locations, suggesting robustness to blocking and censorship.

There is evidence of substantial demand for communication applications that are robust to blocking, censorship, and surveillance but developing such applications requires an understanding of how design decisions influence performance, security, and energy consumption properties. This work takes a first step toward this understanding.

CHAPTER VI

Conclusions

In this thesis we exploit the idea of connecting people's commodity mobile devices, e.g., smartphones and laptops, into distributed Delay Tolerant Networks, in order to provide useful, practical, secure, and censorship-resistant information exchange services to local communities, e.g., a university campus or a town. In particular, our research focuses on studying the utility and practicality, as well as robustness and security of these networks.

6.1 Utility and Practicality

Based on simulation results with real human mobility and contact traces involving 119,055 mobile devices on a university campus that we collected, the largest device connectivity traces to the best of our knowledge, we found **it is practical to use DTNs composed of commodity mobile devices to exchange non-time-sensitive information within local communities, e.g., blog articles and photos**.

Utility: It takes a median of 10.9 hours to exchange information between user pairs in a large-scale DTN covering a university town (the University of Michigan, Ann Arbor). The delays follow a power-law distribution and vary from minutes to a few days.

Although these numbers are far larger than what we usually expect on the Internet, i.e., a few milliseconds, direct comparison between this network's message delivery delay distribution and user response delay distributions of various popular online applications shows that this network could delivery most photos to Flickr users before they "like" the photos, and most blog articles to webblog users before they link the articles. However, it cannot push Tweets quick enough to beat Twitter users' reply delays. To summarize, these networks are useful for exchanging information that is not time sensitive, e.g., blog articles and photos.

Practicality: The energy overhead for participating in this large scale DTN is moderate, e.g., supporting text messages consumes about 14% of a typical smartphone battery in 14 hours.

There are some limitations regarding our evaluation methodology and interesting discoveries that may lead to promising future work,

- Our contact traces are extracted from WiFi association record using the following heuristics common in the DTN research community—devices that are associated with the same WiFi access points have direct connectivity. This may result in connections that do not exist. More importantly, this method cannot capture connections between devices that are associated with different access points. As a result, large connected regions covered by multiple access points may be artificially fragmented into various smaller regions. It is unclear how much this deviation influences our performance estimation and whether correcting it will greatly improve the network performance.

- We used a flooding protocol to analyze the best message delivery delay performance of the network, assuming unrealistically unlimited bandwidth. Understanding the bandwidth capacity of this network, i.e., what is the maximum average bandwidth that each device could inject new information into the network without causing circuitous routes, is important to deepen our understanding of the practicality of DTN systems.

- We have found surprising consistency between the performances of the two datasets we collected, the 1am dataset and the U-M dataset, with the former containing a subset of devices being around 1% of the devices in the latter dataset. On the other hand, the 1am dataset performs very differently from a randomly sampled 1% subset of the U-M dataset. This suggests structural similarities of human communities in different scales that may lead to interesting future work.

- Although it is practical to support text messages on the proposed networks, image messages cost 3.09% of a typical smartphone battery per hour, translating into 42% of the battery in 14 hours. Video messages cost 15.9% per hour, which is even more unacceptable. More energy efficient routing protocols than epidemic flooding are needed to support the last two message types.

6.2 Robustness and Security

Based on simulation results with the large scale device connectivity traces that we collected, we have confirmed that **DTNs composed of commodity mobile devices are**

robust to random node failures and intentional censorship or blocking attacks.

Robustness: Attacks that randomly remove 90% of the network participants only reduce delivery rates to the remaining participants by less than 10%. Targeted attacks, which remove nodes that have most contacts, were more harmful, causing the delivery rate to decrease precipitously when 60% of the participants were removed. However, even when subjected to targeted attacks, the network only suffered a less than 10% decrease in delivery rate when 40% of its participants were removed.

Although structurally robust, the openness of the proposed network introduces numerous security concerns since participants are not vetted. Ensuring security is further complicated because the system is distributed, without any central point of trust. The Sybil attack, for example, is an especially dangerous attack in distributed systems as it breaks the assumption underlying majority voting. **Our work on defeating the Sybil attack lays a foundation for designing security mechanisms on DTNs composed of commodity mobile devices without requiring mutual trust between participating devices**. In particular,

Security: We designed the Mason test, a practical protocol for Sybil defense in wireless ad hoc networks of nodes without mutual trust. The Mason test provides full separation of Sybil and non-Sybil identities, under the condition that conforming nodes outnumber physical attacking nodes.

The robustness and security of the proposed network relies on an unspoken assumption that it is difficult to gain access and control of people's smartphones and laptops in large scale. However, large scale malware installations may be possible, especially given the current monopoly in mobile OS and hardware manufacturers. Depending on specific purposes and thread models of upper-layer applications, secure system designs and implementations require further efforts from application designers, as well as the open source community.

6.3 Prospect

Despite the progress we made in this thesis in confirming the practicality and robustness of infrastructureless networks comprising commodity mobile devices, the lack of a killer application remains the largest barrier to the launch of these networks. FireChat was the most successful application on infrastructureless networks in recent years, but its usage seems to have silently faded after the "Umbrella Revolution". We encourage interested researchers and practitioners to continue the search of useful and influential applications.

APPENDIX

APPENDIX A

Extending Channel Comparison Based Sybil Detection to MIMO Systems

A.1 Introduction

Wireless channel comparison based Sybil defense techniques have been shown effective [1, 2, 74, 79, 131]. By comparing wireless channel conditions one may determine whether two (or more) transmissions originated from the same location and are thus likely to be part of a Sybil attack. The technique is based on the location uniqueness of wireless channels, i.e., even slightly different transmission locations experience uncorrelated wireless channel conditions [72].

Wireless channel comparison based Sybil defenses fall into the class of resource testing based Sybil defenses [63, 64]. In this case transmitter antennas are the resource. Specifically, a group of Sybil transmissions are detected because they demonstrate much less use of antennas (and thus much less wireless channel variations) than the same number of protocol-compliant transmissions.

Existing channel comparison techniques assume single input, single output (SISO) radio systems [1, 2, 74, 79, 131]. In this report we make an initial attempt to extend them to multi-input multi-output (MIMO) systems, which are becoming increasingly popular [80, 132]. Our approach also follows resource testing; we examine received signals to identify transmissions that demonstrate inadequate resource use, or more precisely, to identify numerous transmissions from the same device that claim to be coming from different identities.

Before going into technical details, we will present a mathematical model for MIMO systems and give an intuitive explanation for our technique.

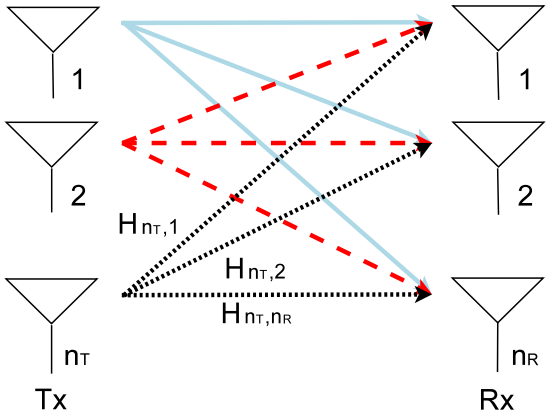


Figure A.1: A MIMO system with n_T transmitters and n_R receivers.

A.1.1 Mathematical Model of MIMO Systems

Figure A.1 illustrates a general MIMO system with n_T transmitters and n_R receivers. Note that the transmitters and receivers may or may not belong to the same MIMO antenna array. Assuming flat-fading or narrowband channels—which is a common practice, this system is modeled as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}. \tag{A.1}$$

All numbers here are in complex space. The n_T-dimensional vector $\mathbf{x}$ represents the transmitted signals and the n_R-dimensional vector $\mathbf{y}$ represents the received signals. $\mathbf{n}$ is the additive white noise. $\mathbf{H} \in \mathbb{C}^{n_R \times n_T}$ is the channel matrix, where $\mathbf{H}_{i,j}$ represents the channel from the jth transmitter to the ith receiver [132]. Note that $\mathbf{H}$ characterizes the wireless channels between the transceivers and is purely determined by the surrounding environment.

Equation A.1 can be expanded as

$$\mathbf{y} = \sum_{j=1}^{n_T} \mathbf{h}_j x_j + \mathbf{n}, \tag{A.2}$$

showing that the received signals of a given MIMO system fall in the column space of its channel matrix $\mathbf{H}$. Since the channel matrix $\mathbf{H}$ is determined by the environment and a Sybil attacker can only change the transmitted signals (x_js) to create Sybil transmissions, this means all these Sybil transmissions fall into the same dimension n_T (at maximum) linear subspace of $\mathbb{C}^{n_R}$. On the other hand, transmissions originated from different MIMO devices fall into different linear subspaces because the corresponding channel matrices vary due to the spatial variations among wireless channels. We thus could decide whether a group of transmissions are originated from the same MIMO system (and thus Sybil) by observing the linear subspace they span. This is the basic observation of our Sybil detection technique.

In the remaining parts of the appendix, we first establish a basic assumption for channel matrix **H** in Section A.2. We then proceed to tackle the particular Sybil detection problem. Section A.3 formally sets up the problem. Section A.4 then develops the observations and techniques for Sybil detection. Finally, Section A.5 discusses practical considerations and future work.

A.2 Channel Matrix Properties

We have shown that the channel matrix **H** plays a critical role in the formation of received signals. Viewing each element $\mathbf{H}_{i,j}$ (i.e., the wireless channel from the jth transmitter to the ith receiver) as a random variable, Conjecture 1 establishes a basic assumption for **H**.

Conjecture 1. The channel matrix of a MIMO system has full rank if its wireless channels are all independently distributed.

According to existing results, the probability that a square random matrix with all independent entries is singular quickly approaches zero when its size increases, and equals zero asymptotically [133, 134]. Conjecture 1 neglects this small probability and assumes a non-singular matrix for finite size random matrices, mainly for the ease of theoretical development. The rare singular cases do not invalidate our method, but do affect the performance negatively. We will leave the analysis to future experimental evaluations.

A.3 Problem Setup

A set of MIMO devices observe the wireless transmissions from another set of MIMO devices. The observing devices have n_R receiver antennas in total. There are two type of devices. Conforming devices only make one transmission each; we call these transmissions non-Sybil transmissions. Attacking devices make more than one transmissions each, and we call these transmissions Sybil transmissions. Our goal is to detect all the Sybil transmissions.

It is important to distinguish among two types of Sybil transmissions. If the number of Sybil transmissions an attacker makes is less than or equal to its number of transmitter antennas, then these Sybil transmissions are called *undetectable Sybil transmissions*. Otherwise, if the number of Sybil transmissions an attacker makes is greater than its number of transmitter antennas, these Sybil transmissions are called *detectable Sybil transmissions*. It will become evident in Section A.4 that we are only able to identify detectable Sybil transmissions.

The following conditions are necessary for a wireless channel comparison based Sybil defense to work.

Condition 4. An attacking device is stationary during its Sybil transmissions.

Condition Condition 4 is necessary because free device motion allows Sybil transmissions to experience different wireless channels. It is implicitly assumed by all previous channel comparison based Sybil defenses. We proposed a method to detect device motion, in case restricting it is impossible (see Section 2.7).

Condition 5. The total number of receivers n_R is larger than the number of transmitters on any single attacking device.

Condition Condition 5 upper bounds the number of transmitters per device by n_R. N_T denotes the maximum number of transmitters per attacking device.

The following assumptions create an ideal world to simplify explanation and theoretical development.

1. All the wireless channels in the MIMO system are independent.

2. There is no noise in the system, i.e., $\mathbf{n} = \mathbf{0}$ in Equation A.1 and Equation A.2.

In reality both assumptions can be relaxed. For the first one, our method works as long as wireless channels from different MIMO devices are independent, which is commonly true thanks to the rich spatial variations of wireless channels [72]. For the second one, we leave it to future experiments to quantify the actual noise threshold.

A.4 Sybil Detection

Section A.1 suggested that we could detect a group of Sybil transmissions by observing that they demonstrate much less wireless channel variations and thus fewer transmitter antennas than a group of non-Sybil transmissions with equal size. In this section we detail the properties that allow us to make this distinction.

Lemma 6. *The received signals of any group of no more than $\lfloor \frac{n_R}{N_T} \rfloor$ non-Sybil transmissions are linearly independent.*

Proof. Use n to denote the total number of transmitters involved. $n \leq n_R$ because the total number of transmissions is no more than $\lfloor \frac{n_R}{N_T} \rfloor$ and each of them involves N_T transmitters at maximum.

Use $\mathbf{H}$ to represent a MIMO system with all these transmissions' transmitters and the given receivers. $\mathbf{H}$ can be written as

$$\mathbf{H} = \begin{pmatrix} \mathbf{h}_1 & \mathbf{h}_2 & ... & \mathbf{h}_n \end{pmatrix},$$

where $\mathbf{h}_i$ denotes the vector of wireless channels from the i-th transmitter to all the n_R receivers.

We now prove $\mathbf{h}_i$s are linearly independent. Since all the wireless channels in $\mathbf{H}$ are independent (see the first assumption in Section A.3), according to Conjecture 1 $\mathbf{H}$ has full rank. Furthermore, since $\mathbf{H}$'s column size is smaller than or equal to its row size ($n \leq n_R$), the columns of $\mathbf{H}$ are linearly independent.

Finally, the received signals are linearly independent because they are linear combinations of $\mathbf{h}_i$s. □

Corollary 1. It requires (the received signals of) at least $\lfloor \frac{n_R}{N_T} \rfloor$ non-Sybil transmissions to linearly expand (the received signal of) a non-Sybil transmission.

Corollary 1 is just a restatement of Lemma Lemma 6. Because a group of Sybil transmissions only correspond to less (or equal) independent channel vectors than a group of non-Sybil transmissions of the same size, Corollary 1 can be generalized as follows.

Corollary 2. It requires (the received signals of) at least $\lfloor \frac{n_R}{N_T} \rfloor$ transmissions to linearly expand (the received signal of) a non-Sybil transmission.

So far Corollary 2 describes the features of non-Sybil transmissions. On the other hand, the following lemma describes the features of Sybil transmissions.

Lemma 7. *For any detectable Sybil transmission, there always exists a group of transmissions with size no more than N_T, whose received signals could linearly expand (the received signal of) that transmission.*

Proof. Since all Sybil transmissions from a same device fall into a linear subspace of dimension n_T (see Equation A.2 in Section A.2) and there are more than n_T detectable Sybil transmissions by definition, any group of $n_T + 1$ detectable Sybil transmissions are linearly dependent. In other words, any group of n_T ($n_T < N_T$) detectable Sybil transmissions from the same device could linearly expand a detectable Sybil transmission. □

Corollary 2 and Lemma Lemma 7 indicate that the size of linearly expanding transmission groups can be a distinguishing feature of Sybil vs. no-Sybil classifications, under a stricter condition of maximum number of transmitters per device.

Condition 6. $N_T < \lfloor \frac{n_R}{N_T} \rfloor$

Lemma 8. *A transmission is detectable Sybil, if and only if there exists a group of less than $\lfloor \frac{n_R}{N_T} \rfloor$ transmissions, whose received signals could linearly expand (the received signal of) that transmission.*

Proof. It is easy to see this is true under Condition Condition 6, according to Corollary 2 and Lemma Lemma 7. □

Note that undetectable Sybil transmissions may or may not be identified. However, from the viewpoint of resource testing, these transmissions are not disproportional to the attacking device's resources (i.e., transmitter antennas) and therefore benign.

A.4.1 Naive Algorithm

Lemma Lemma 8 leads to a naive Sybil detection algorithm. Produce all possible combinations of $(\lfloor \frac{n_R}{N_T} \rfloor - 1)$-tuples of transmissions and examine them one by one. For a given tuple, use it to examine all the remaining transmissions; ones that can be linearly expanded by the current tuple are marked as Sybil.

This algorithm allows us to identify all detectable Sybil transmissions because all possible combinations of $(\lfloor \frac{n_R}{N_T} \rfloor - 1)$-tuples are considered. A direction for future improvements is to intelligently reduce the possible tuples in consideration while maintaining the same level of false negative rate.

A.5 Practical Consideration and Future Work

Our current solution assumes complete information about the received signals, i.e., both the power and phase of the received signal at each receiver of the MIMO antenna. In practice, we may only know the received power, not the phase. Worse, we may only know the aggregated power of all receivers if only an aggregated RSSI is exposed. A path of future work is to relax the current assumption and push the work further to deal with incomplete information.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] 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.

[2] 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.

[3] D. C. Weiss, "NSA 'systematically violated' safeguards for phone surveillance program, declassified opinion says," *ABA Journal*, Sep. 11 2013.

[4] B. Mears and E. Perez, "Judge: NSA domestic phone data-mining unconstitutional," *CNN.com*, Dec. 17 2013.

[5] "Report: Feds getting phone records of all Verizon customers," *CBS News*, Jun. 12 2013.

[6] 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, Jan. 2011.

[7] 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.

[8] Reporters Without Borders, "Enemies of the internet report 2012," pp. 1–71, Mar. 2012.

[9] Reporters Without Borders, "Enemies of the internet report 2013," pp. 1–47, Mar. 2013.

[10] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *Proc. SIGCOMM Computer Communication Review*, vol. 29, no. 4, 1999, pp. 251–262.

[11] W. Willinger, D. Alderson, and J. C. Doyle, "Mathematics and the Internet: A source of enormous confusion and great potential," *Notices of the American Mathmatical Society*, no. 5, pp. 586–599, 2009.

[12] R. Albert, H. Jeong, and A.-L. Barabási, "Error and attack tolerance of complex networks," *Nature*, vol. 406, no. 6794, pp. 378–382, 2000.

[13] "comScore releases April 2014 U.S. search engine rankings," comScore, May 2014, http://www.comscore.com/Insights/Market-Rankings/comScore-Releases-April-2014-US-Search-Engine-Rankings.

[14] S. Kent and K. Seo, "RFC4301: Security architecture for the internet protocol," RFC 4301, Internet Engineering Task Force, Dec. 2005. [Online]. Available: http://tools.ietf.org/html/rfc4301

[15] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol, version 1.2," RFC 5246, Internet Engineering Task Force, Aug. 2008. [Online]. Available: http://tools.ietf.org/html/rfc5246

[16] "Gnupg," https://www.gnupg.org/.

[17] 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.

[18] "Silent Text," https://silentcircle.com/services.

[19] "Cryptocat," https://crypto.cat/.

[20] R. Singel, "Encrypted e-mail company Hushmail spills to feds," *Wired.com*, Nov. 07 2007, http://www.wired.com/2007/11/encrypted-e-mai/.

[21] H. Hoogstraaten, R. Prins, D. Higgebrugge, D. Heppener, F. Groenewegen, J. Wettinck, K. Strooy, P. Arends, P. Pols, R. Kouprie, S. Moorrees, X. van Pelt, and Y. Z. Hu, "Black tulip: Report of the investigation into the DigiNotar certificate authority breach," Fox-IT, Tech. Rep., Aug. 2012.

[22] P. R. Zimmermann, *The official PGP user's guide*. MIT Press, 1995.

[23] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: the second-generation onion router," in *Proc. USENIX Security Symp.*, Aug. 2004, p. 21.

[24] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A distributed anonymous information storage and retrieval system," in *Proc. Int. Wkshp. Design Issues in Anonymity and Unobservability*, July 2000, pp. 46–66.

[25] "RetroShare," http://retroshare.sourceforge.net/index.html.

[26] K. Gallagher, "How Tor helped catch the Harvard bomb threat suspect," *The Daily Dot*, Dec. 18 2013, http://www.dailydot.com/crime/tor-harvard-bomb-suspect/.

[27] G. Fanti, Y. Ben David, S. Benthall, E. Brewer, and S. Shenker, "Rangzen: Circumventing government-imposed communication blackouts," University of California, Berkeley, Tech. Rep. UCB/EECS-2013-128, July 2013.

[28] "freifunk.net," http://freifunk.net/.

[29] "Project Meshnet," https://projectmeshnet.org/.

[30] "Open Garden: our family of apps," https://opengarden.com/apps.

[31] A. Lindgren and P. Hui, "The quest for a killer app for opportunistic and delay tolerant networks," in *Proc. Wkshp. Challenged Networks*, Sept. 2009, pp. 59–66.

[32] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *Proc. Wkshp. on Delay-tolerant Networking*, Aug. 2005, pp. 244–251.

[33] 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.

[34] J. Cook, "Hong Kong protesters are communicating via a mobile app that doesn't actually use the Internet," *businessinsider.com*, Sept. 2014.

[35] E. M. Royer and C.-K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks," *Personal Communications, IEEE*, vol. 6, no. 2, pp. 46–55, 1999.

[36] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. SIGCOMM Computer Communication Review*, Aug. 2003, pp. 27–34.

[37] C. P. Mayer and O. P. Waldhorst, "Routing in hybrid delay tolerant networks," *Computer Communications*, vol. 48, pp. 44–55, July 2014.

[38] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19–20, 2003.

[39] E. M. Daly and M. Haahr, "Social network analysis for information flow in disconnected delay-tolerant MANETs," *IEEE Trans. Mobile Computing*, vol. 8, no. 5, pp. 606–621, 2009.

[40] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Duke University, Tech. Rep. CS-200006, 2000.

[41] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *IEEE Trans. Mobile Computing*, vol. 6, no. 6, pp. 606–620, 2007.

[42] N. Ristanovic, J.-Y. L. Boudec, A. Chaintreau, and V. Erramilli, "Energy efficient offloading of 3G networks," in *Int. Conf. Mobile Adhoc and Sensor Systems*, Oct. 2011, pp. 202–211.

[43] T. Hossmann, F. Legendre, P. Carta, P. Gunningberg, and C. Rohner, "Twitter in disaster mode: Opportunistic communication and distribution of sensor data in emergencies," in *Proc. Extreme Conf. on Communication: The Amazon Expedition*, Sept. 2011, pp. 1:1–1:6.

[44] N. Eagle and A. S. Pentland, "Reality Mining: Sensing complex social systems," *Personal and Ubiquitous Computing*, vol. 10, no. 4, pp. 255–268, May 2006.

[45] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," *Computer Networks*, vol. 52, no. 14, pp. 2690–2712, 2008.

[46] M. McNett and G. M. Voelker, "Access and mobility of wireless PDA users," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 2, pp. 40–55, 2005.

[47] V. Srinivasan, M. Motani, and W. T. Ooi, "Analysis and implications of student contact patterns derived from campus schedules," in *Proc. Int. Conf. Mobile Computing and Networking*, Sept. 2006, pp. 86–97.

[48] T. Karagiannis, J.-Y. L. Boudec, and M. Vojnovic, "Power law and exponential decay of intercontact times between mobile devices," *IEEE Trans. Mobile Computing*, vol. 9, no. 10, pp. 1377–1390, Oct. 2010.

[49] T. Hossmann, T. Spyropoulos, and F. Legendre, "A complex network analysis of human mobility," in *Conf. on Computer Communications Workshops*, Apr. 2011, pp. 876–881.

[50] T. Hossmann, T. Spyropoulos, and F. Legendre, "Putting contacts into context: mobility modeling beyond inter-contact times," in *Proc. Int. Symp. Mobile Ad Hoc Networking and Computing*, May 2011, pp. 18:1–18:11.

[51] N. Sastry, D. Manjunath, K. Sollins, and J. Crowcroft, "Data delivery properties of human contact networks," *IEEE Trans. Mobile Computing*, vol. 10, no. 6, pp. 868–880, June 2011.

[52] E. Yoneki, P. Hui, S. Chan, and J. Crowcroft, "A socio-aware overlay for publish/subscribe communication in delay tolerant networks," in *Proc. Symp. Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, Oct. 2007, pp. 225–234.

[53] R. Groenevelt, P. Nain, and G. Koole, "The message delay in mobile ad hoc networks," pp. 210–228, 2005.

[54] A. Picu, T. Spyropoulos, and T. Hossmann, "An analysis of the information spreading delay in heterogeneous mobility DTNs," in *IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks*, June 2012, pp. 1–10.

[55] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance modeling of epidemic routing," pp. 2867–2891, 2007.

[56] Z. J. Haas and T. Small, "A new networking model for biological applications of ad hoc sensor networks," pp. 27–40, 2006.

[57] Y. Liu, D. R. Bild, D. Adrian, G. Singh, R. P. Dick, D. S. Wallach, and Z. M. Mao, "Performance and Energy Consumption Analysis of a Delay-Tolerant Network for Censorship-Resistant Communications," in *Proc. Int. Symp. on Mobile Ad Hoc Networking and Computing*, June 2015, pp. 257–266.

[58] M. C. González, C. A. Hidalgo, and A.-L. Barabási, "Understanding individual human mobility patterns," *Nature*, vol. 453, pp. 778–782, June 2008.

[59] A. Noulas, S. Scellato, R. Lambiotte, M. Pontil, and C. Mascolo, "A tale of many cities: universal patterns in human urban mobility," *PLOS ONE*, vol. 7, no. 5, p. e37027, May 2012.

[60] "1am: Censorship-resistant microblogging," 2013, http://1am-networks.org.

[61] 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.

[62] P. Gardner-Stephen, "Sustaining telecommunications capability and capacity during acute phase of disasters and disaster responses," *Prehospital and Disaster Medicine*, vol. 26, pp. s94–s95, May 2011.

[63] J. Douceur, "The Sybil attack," in *Proc. Int. Wkshp. Peer-to-Peer Systems*, Mar. 2002, pp. 251–260.

[64] 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.

[65] 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, Tech. Rep. 2006-052, Oct. 2006.

[66] H. Zhou, M. Mutka, and L. Ni, "Multiple-key cryptography-based distributed certificate authority in mobile ad-hoc networks," in *Proc. Global Telecommunications Conf.*, vol. 3, Nov. 2005, pp. 1681–1685.

[67] 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.

[68] N. Borisov, "Computational puzzles as Sybil defenses," in *Proc. Int. Conf. Peer-to-Peer Computing*, Sept. 2006, pp. 171–176.

[69] F. Li, P. Mittal, M. Caesar, and N. Borisov, "SybilControl: Practical Sybil defense with computational puzzles," in *Proc. Wkshp. Scalable Trusted Computing*, Oct. 2012.

[70] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: defending against Sybil attacks via social networks," in *Proc. SIGCOMM Computer Communication Review*, Sept. 2006, pp. 267–278.

[71] 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.

[72] T. S. Rappaport, *Wireless Communications: Principles & Practice*. Prentice-Hall, NJ, 2002.

[73] 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.

[74] 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.

[75] 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.

[76] T. Suen and A. Yasinsac, "Peer identification in wireless and sensor networks using signal properties," in *Proc. Int. Conf. Mobile Ad hoc and Sensor Systems*, Nov. 2005, pp. 826–833.

[77] S. Lv, X. Wang, X. Zhao, and X. Zhou, "Detecting the Sybil attack cooperatively in wireless sensor networks," in *Proc. Int. Conf. Computational Intelligence and Security*, Dec. 2008, pp. 442–446.

[78] M. S. Bouassida, G. Guette, M. Shawky, and B. Ducourthial, "Sybil nodes detection based on received strength variations within VANET," *Int. J. Network Security*, vol. 9, no. 1, pp. 22–33, July 2009.

[79] 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.

[80] 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.

[81] 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.

[82] 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.

[83] A. T. Welford and J. M. T. Brebner, *Reaction Times*. Academic Press, 1980.

[84] "Using Wi-Fi P2P for service discovery," http://developer.android.com/training/connect-devices-wirelessly/nsd-wifi-direct.html.

[85] M. S. Gast, *802.11 Wireless Networks: The Definitive Guide*, 2nd ed. O'Reilly, Apr. 2005.

[86] C. Benvenuti, *Understanding Linux Network Internals*, 1st ed. O'Reilly Media, Jan. 2006.

[87] "Amazon S3," http://aws.amazon.com/s3/.

[88] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," vol. 41, no. 4, Aug. 2010, pp. 159–170.

[89] "Net Index by OOKLA," http://www.netindex.com/download/2,1/United-States/.

[90] "Project voldemort," http://www.project-voldemort.com/voldemort/.

[91] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11 b mesh network," vol. 34, no. 4, Aug. 2004, pp. 121–132.

[92] K. Chintalapudi, A. P. Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *Proc. Int. Conf. Mobile Computing and Networking*, Sept. 2010, pp. 173–184.

[93] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," in *Proc. Int. Conf. Mobile Computing and Networking*, Aug. 2012, pp. 269–280.

[94] T. S. Rappaport, *Wireless Communications: Principles & Practice*. Prentice-Hall, NJ, 1996.

[95] L. Song and D. Kotz, "Evaluating opportunistic routing protocols with large realistic contact traces," in *Proc. Wkshp. Challenged Networks*, Sept. 2007, pp. 35–42.

[96] D. R. Bild, Y. Liu, R. P. Dick, Z. M. Mao, and D. S. Wallach, "Aggregate characterization of user behavior in Twitter and analysis of the retweet graph," *ACM Trans. Internet Technologies*, 2015.

[97] F. Putze, P. Sanders, and J. Singler, "Cache-, hash-, and space-efficient Bloom filters," *J. of Experimental Algorithms*, no. 4, pp. 108–121, 2007.

[98] S. Trifunovic, B. Distl, D. Schatzmann, and F. Legendre, "WiFi-Opp: ad-hoc-less opportunistic networking," in *Proc. Wkshp. Challenged Networks*, Sept. 2011, pp. 37–42.

[99] "Manes: a mobile ad hoc network emulation system," 2013, http://ziyang.eecs.umich.edu/projects/whisper/manes/.

[100] D. Wang, D. Pedreschi, C. Song, F. Giannotti, and A.-L. Barabasi, "Human mobility, social ties, and link prediction," in *Proc. Int. Conf. Knowledge Discovery and Data Mining*, Aug. 2011, pp. 1100–1108.

[101] "University of Michigan—total enrollment overview," 2015, http://www.ro.umich.edu/report/15enrollmentsummary.pdf.

[102] "University of Michigan—staff headcounts by gender, race/ethnicity, and job family," 2015, http://obp.umich.edu/wp-content/uploads/pubdata/factsfigures/stafffed+max_umaa+hosp+all_fall14.pdf.

[103] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow, "The anatomy of the facebook social graph," 2011.

[104] 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.

[105] R. Pastor-Satorras, A. Vazquez, and A. Vespignani, "Dynamical and correlation properties of the internet," *Physical Review Letters*, p. 258701, 2001.

[106] 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

[107] M. E. Newman, "Mixing patterns in networks," *Physical Review E*, p. 026126, 2003.

[108] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proc. Internet Measurement Conf.*, 2007, pp. 29–42.

[109] P. Erdos and A. Renyi, "On random graphs i." *Publicationes Mathematicae*, pp. 290–297, 1959.

[110] B. Ries, "Twitter blocks pro-Ukrainian political account for Russian users," *Mashable.com*, May 2014.

[111] A. Al-Akkad, C. Raffelsberger, A. Boden, L. Ramirez, and A. Zimmermann, "Tweeting when online is off? Opportunistically creating mobile ad-hoc networks in response to disrupted infrastructure," in *Proc. Int. Conf. Information Systems for Crisis Response and Management*, May 2014, pp. 662–671.

[112] A. Socievole and S. Marano, "Evaluating the impact of energy consumption on routing performance in delay tolerant networks," in *Proc. Int. Conf. Wireless Communications and Mobile Computing*, Aug. 2012, pp. 481–486.

[113] Y. Li, Y. Jiang, D. Jin, L. Su, L. Zeng, and D. O. Wu, "Energy-efficient optimal opportunistic forwarding for delay-tolerant networks," *IEEE Trans. Vehicular Technology*, vol. 59, no. 9, pp. 4500–4512, 2010.

[114] X. Lu and P. Hui, "An energy-efficient n-epidemic routing protocol for delay tolerant networks," in *Proc. Int. Conf. Networking, Architecture and Storage*, July 2010, pp. 341–347.

[115] E. Bakshy, I. Rosenn, C. Marlow, and L. Adamic, "The role of social networks in information diffusion," Apr. 2012, pp. 519–528.

[116] 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.

[117] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst, "Cascading behavior in large blog graphs: patterns and a model," Carnegie Mellon University, Tech. Rep. CMU-ML-06113, 2006.

[118] E. Bayraktaroglu, C. King, X. Liu, G. Noubir, R. Rajaraman, and B. Thapa, "Performance of IEEE 802.11 under jamming," *Mobile Networks and Applications*, vol. 18, no. 5, pp. 678–696, 2013.

[119] Monsoon, "Monsoon power monitor," 2008, http://www.msoon.com/LabEquipment/PowerMonitor/.

[120] R. Zheng, J. C. Hou, and L. Sha, "Asynchronous wakeup for ad hoc networks," in *Proc. Int. Symp. Mobile Ad Hoc Networking and Computing*, June 2003, pp. 35–45.

[121] A. M. Braga and E. N. Nascimento, "Portability evaluation of cryptographic libraries on Android smartphones," in *Proc. Int. Symp. Cyberspace Safety and Security*, Dec. 2012, pp. 459–469.

[122] J. Lopéz and R. Dahab, "An overview of elliptic curve cryptography," Institute of Computing, State University of Campinas, Tech. Rep. IC–00-10, 2000.

[123] "Sponge Castle," 2014, http://rtyley.github.io/spongycastle/.

[124] "OpenSSL," 2014, https://www.openssl.org/.

[125] "Twitter usage," 2015, https://about.twitter.com/company.

[126] "A focus on efficiency, a whitepaper from Facebook, Ericsson and Qualcomm," internet.org, Sept. 2013.

[127] ComScore, "ComScore releases December 2013 U.S. online video rankings," *comscore.com*, Jan. 2014.

[128] "YouTube statistics," 2014, https://www.youtube.com/yt/press/statistics.html.

[129] Y. Sasson, D. Cavin, and A. Schiper, "Probabilistic broadcast for flooding in wireless mobile ad hoc networks," in *Wireless Communications and Networking*, vol. 2, Mar. 2003, pp. 1124–1130.

[130] A. M. Hanashi, I. Awan, and M. Woodward, "Performance evaluation with different mobility models for dynamic probabilistic flooding in manets," *Mobile Information Systems*, no. 1, pp. 65–80, 2009.

[131] 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.

[132] Q. H. Spencer, J. W. Wallace, C. B. Peel, T. Svantesson, A. L. Swindlehurst, H. Lee, and A. Gumalla, "Performance of multi-user spatial multiplexing with measured channel data," in *MIMO System Technology for Wireless Communications*, G. Tsoulos, Ed. CRC Press, 2006.

[133] M. Rudelson and R. Vershynin, "The Littlewood-Offord problem and invertibility of random matrices," *Advances in Mathematics*, vol. 218, pp. 600–633, June 2008.

[134] G. Pan and W. Zhou, "Circular law, extreme singular values and potential theory," May 2007, `arXiv:0705.3773 [math.PR]`.