SMILE: Encounter-Based Trust for Mobile Social Services

Justin Manweiler
Duke University
Durham, NC, USA
jgm@cs.duke.edu

Ryan Scudellari
Duke University
Durham, NC, USA
scudella@cs.duke.edu

Landon P. Cox
Duke University
Durham, NC, USA
lpcox@cs.duke.edu

ABSTRACT

Conventional mobile social services such as Loopt and Google Latitude rely on two classes of trusted relationships: participants trust a centralized server to manage their location information and trust between users is based on existing social relationships. Unfortunately, these assumptions are not secure or general enough for many mobile social scenarios: centralized servers cannot always be relied upon to preserve data confidentiality, and users may want to use mobile social services to establish new relationships. To address these shortcomings, this paper describes SMILE, a privacy-preserving "missed-connections" service in which the service provider is untrusted and users are not assumed to have pre-established social relationships with each other. At a high-level, SMILE uses short-range wireless communication and standard cryptographic primitives to mimic the behavior of users in existing missed-connections services such as Craigslist: trust is founded solely on anonymous users' ability to prove to each other that they shared an encounter in the past. We have evaluated SMILE using protocol analysis, an informal study of Craigslist usage, and experiments with a prototype implementation and found it to be both privacy-preserving and feasible.

Categories and Subject Descriptors

C.2.0 [**General**]: Security and protection

General Terms

Design, Security

Keywords

Location-based Services, Privacy, Trust, Social Networking

CCS'09, November 9–13, 2009, Chicago, Illinois, USA.

1. INTRODUCTION

Programmable consumer devices such as mobile phones have placed computation within arm's reach at all times and in all places. *Mobile social services* take advantage of the nearly constant physical proximity of devices to their owners to enable a wide range of new social interactions. In a conventional mobile social service, devices send intermittent location updates to a service provider, which uses those locations to coordinate interactions among participants. For example, Google Latitude [7] and Loopt [20] are popular services that allow users to share their location information with friends.

Within existing mobile social services, trust is founded on two classes of relationships: one with the service provider and another with peers. Service providers are treated as benevolent guardians of their location data. Users control which participants can track their location, but their location privacy is not protected from the service providers themselves. Trust between users is almost always based on pre-established social relationships. Social groups, such as work colleagues, family members, and friends, typically define a subset of users that may access a participant's location information.

Unfortunately, neither class of trust relationship provides a secure or general foundation for mobile social services. First, lessons from existing online social networks (OSNs) demonstrate the many ways that data confidentiality can be compromised by trusted service providers: users' sensitive data can be inadvertently leaked [21], can fall under the control of hackers [15], and can be abused by service administrators [25]. The potential leakage of users' long-term location histories is a serious threat, and would be more damaging than leaks of the media and messaging state currently managed by OSNs. In addition, restricting location sharing to pre-established social relations makes a large class of compelling mobile social services impossible. For example, services such as Social Serendipity [6], which notifies users when like-minded strangers are nearby, are impossible if all trust relationships must be pre-established.

As a result, this paper describes SMILE, a mobile social service in which trust is established solely on the basis of shared encounters; the service provider is not trusted to access users' location information and we assume no pre-established trust relationships among users. At the heart of the service is the notion of an *encounter*, which is defined as a short period of co-location between people. Our service is modeled after the popular "missed-connections" services found in newspapers and websites like Craigslist.

The key features of a missed-connections service are: (1) strangers who were at the same place and time should be able to contact each other at a later time; (2) once connected, those strangers should be able to prove to each other that they actually

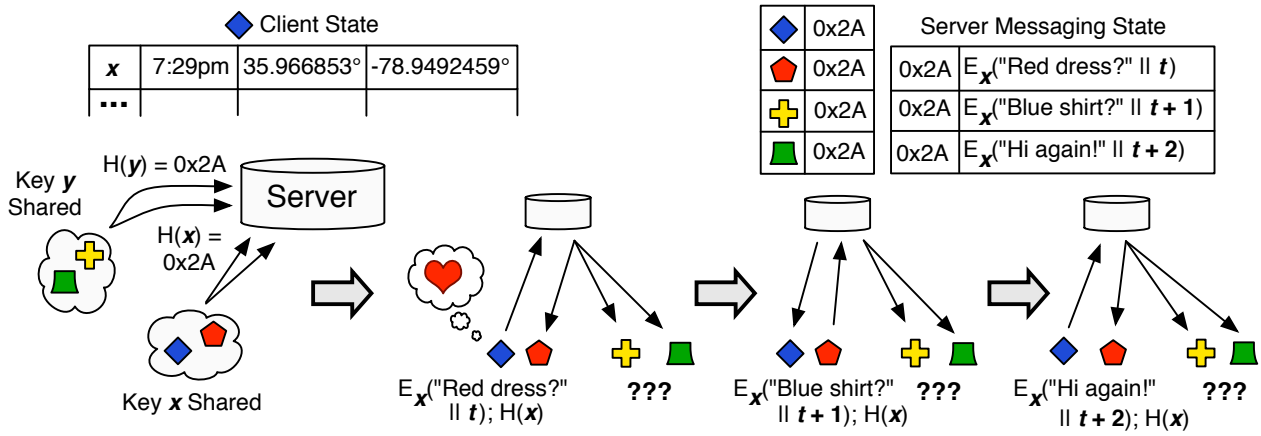

Figure 1: An illustrated sequence of operations. Let H denote a cryptographic hash function and $E_x(m)$ denote the encryption of message m with key x. Encounter keys x and y hash to the same value, leading the server to relay $E_x(m)$ to participants in both encounters. However, only participants with key x can recover message m. A timestamp t nonce in the reply prevents replay attacks.

encountered one another. We use three complementary techniques to provide these features without exposing users' location information to either the service provider or adversaries claiming to have been physically present at a particular place and time:

1. Co-located participants perform periodic *passive key exchange* with each other using short-range wireless broadcasts.
2. Participants use *key hashes* to establish a rendezvous point at a centralized server without exposing the encounter location to the service provider.
3. Participants limit the service provider's ability to infer which pairs of users were involved in an encounter by carefully inducing *key-hash collisions* at the server and relying on clients to resolve ambiguities.

The high-level insight behind these techniques is derived from observations of existing online missed-connections services. In these services, a poster normally asks anonymous respondents to confirm small details from the encounter. For example, to ensure that she is communicating with her waiter from the previous night, a user might ask a respondent to tell her what she ordered. SMILE's passive key exchange protocol functions similarly, by creating shared knowledge about an encounter that only participants could have recorded.

We have evaluated SMILE using protocol analysis, by characterizing behavior within the missed-connections feature of Craigslist, and through experiments with a prototype implementation. Based on this analysis, we have found that SMILE provides users with both location and encounter privacy from adversarial service providers and peers, and that our passive key-exchange protocol is feasible using a widely-deployed, short-range wireless technology, such as Bluetooth.

The rest of the paper is organized as follows: in Section 2, we outline our basic assumptions and threat model; in Section 3, we present a server-centric missed-connections system utilizing collisions in a centralized hash table to provide k-anonymity; in Section 4, we consider an alternative distributed missed-connections scheme relying on an anonymized remailing or onion-routing network; in Section 5, we evaluate the feasibility of our scheme; in Section 6, we present related work; we conclude in Section 7.

2. TRUST AND THREAT MODEL

SMILE allows strangers who shared an *encounter* in the past to communicate at a later point in time. An encounter is defined as two people being in close physical proximity to each other for a period of time. The challenge addressed in this paper is providing a missed-connections service with strong *location-privacy* and *encounter-privacy* guarantees. A user's location privacy is violated when either the service provider or an unauthorized user can infer with high probability that the user was in a particular place at a particular time. Similarly, a user's encounter privacy is violated when the service provider or an unauthorized user can infer with high probability that two users were in the same place at the same time.

Many privacy threats exist independently of SMILE, and are thus beyond the scope of this paper. For example, we make no attempt to conceal devices' locations from cellular-network operators, access-point administrators, or any other snooping radios. Many attacks can be launched from these vantage points due to the pervasive use of static MAC addresses and identifiable traffic patterns in wireless networks. Work on disposable addresses [10], prolonged silent periods [14], and privacy-preserving link-layer protocols [8] offer solutions and could be plugged-in, when available.

Unlike most mobile social services, we do not assume that trust is derived from pre-established social relationships. Instead, trust in SMILE is based only on shared encounters. Assuming there is mutual interest in establishing communication, two users trust each other only if they can convince each other that they were in the same place at the same time. In the absence of mutual interest or proof of an encounter, users remain anonymous to each other.

Adversarial Capabilities. We utilize a central server to aid in post-encouter matching. This infrastructure, and all other third-parties, are considered untrusted. Further, we assume that all adversaries are endowed with at least the following set of capabilities. We assume that servers have access to substantial personal information about all users, including each user's full name, billing address, IP-localized home address, and credit card information. We further assume that an attacker can arbitrarily read or replace user data and network traffic. This allows server administrators to perform timing analysis on user data, forge user data, interpose on communication between users, masquerade as any user, and replay user messages.

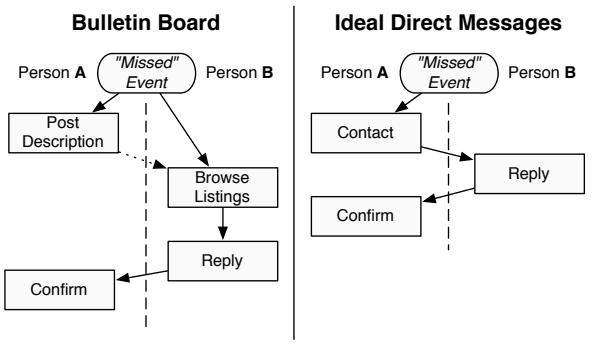

Figure 2: In online missed-connections posting services (such as Craigslist), posting subjects are forced to manually browse up to hundreds of unrelated postings. By directly routing messages to encounter participants, SMILE is more efficient and less error-prone.

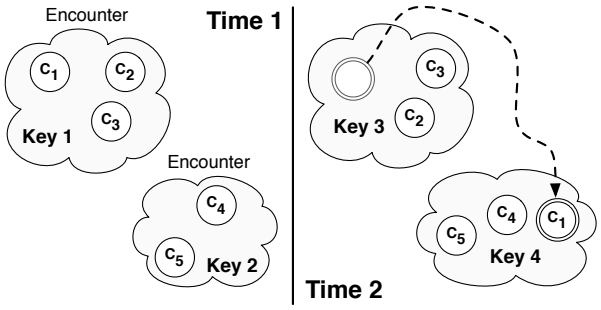

Figure 3: Wireless encounter-key broadcasts provide co-located users with shared state that can later be used to prove participation in an encounter.

Adversarial Limitations. On the other hand, we also assume that malicious participants and servers are limited in the following ways. First, within a "home" geographic region (i.e., a metro area), we assume that all users know approximately how many users participate in the system and how often users register and respond to missed-connections requests. Users could obtain this information out-of-band via a third-party monitoring service such as Alexa [2]. We also assume that participants do not share information about an encounter with users who were not present. Collusion of this form violates our trust model and can allow a user who was not part of an encounter to generate a false proof. Finally, we assume limited collusion among malicious users and service providers. Successful collusion attacks require subversion of a large portion of legitimate system users or an adversary who was physically proximate to an encounter.

3. SMILE SYSTEM DESIGN

In this section, we present the design of SMILE, *Secure MIssed connections through Logged Encounters*. SMILE is a secure, centralized missed-connections service. The basic structure of SMILE's messaging protocol is as follows: (1) mobile users passively exchange cryptographic keys with nearby peers; (2) users periodically upload batches of key hashes to a central, coordinating server; (3) a user sends a message to the server encrypted with one such key and labels it with the corresponding key hash; (4) the server forwards the encrypted message to all users that have uploaded the same key hash; (5) only encounter participants are able to decrypt the message. SMILE offers protection against malicious agents attempting to determine or disclose a user's location history, encounter history, or private messages. Figure 1 presents a high-level depiction of the SMILE protocol.

The bulletin-board approach of traditional posting services and newsprint personals has two primary drawbacks. First, these services require participants in an encounter to actively search for their match. This scheme is inconvenient and inefficient: one person must post a listing and hope that the other will find it after extensive manual browsing. Second, because anyone can respond to a posting, even if they were not present for the encounter, existing services provide very weak authentication guarantees.

Figure 2 shows the "ideal" approach, in which a user's missed-connections messages are routed directly to their intended recipient. Would-be recipients would not be required to search ads or websites to receive a message. Our aim is closely approximate this ideal service without compromising participants' privacy.

3.1 Encounter Detection

Users participate in SMILE by carrying a smartphone, laptop, or other mobile device running a lightweight sensing application. Through short-range wireless communication, participants sense the presence of others in their proximity. An incident of mutual detection is considered an *encounter*. During each encounter, co-located peers use a wireless link to establish a random symmetric key. Hereafter, we will denote this shared state as the *encounter key*. The encounter key may be randomly generated by either party. Figure 3 depicts encounter-key distribution.

Local Encounter State. A user's device automatically logs encounter keys, along with when and where they were received, in a database. Localization need not be highly precise, and can be determined using GPS, WiFi access points, or GSM towers. Locations only help the user identify or recall past encounters. The privacy risk of recording this information is minimal, as it will never be provided to the server or any peer. The local database may be encrypted against a password for protection in case of device theft.

Encounter Provenance. If other useful state is also available, it may be stored as a supplement to time and location. The aggregation of this state amounts to a record of human-interpretable *encounter provenance*, metadata describing the origin of the encounter record. This idea is similar in spirit to provenance systems proposed for data storage [28] and web browsers [23]. For example, a smartphone could record what websites the user was browsing, active chat sessions, the most recently dialed telephone number, the last photograph taken, or the song playing at the time.

A sophisticated device could be even more proactive in recording provenance state. For example, if equipped with an accelerometer, the device could perform activity recognition (e.g., biking, jogging, walking, or sitting) [3]. The device could also automatically take a picture or series of pictures whenever an encounter occurs. Face detection software, which is already present on many consumer point-and-shoot cameras, could then be used to filter photographs that do not identify the surrounding people during an encounter. Finally, short audio recordings, taken at the time of the encounter [1], could provide an audio context for an encounter. We leave a more in-depth discussion of these techniques for future work. For the rest of the paper, we assume that local records include only location and time information.

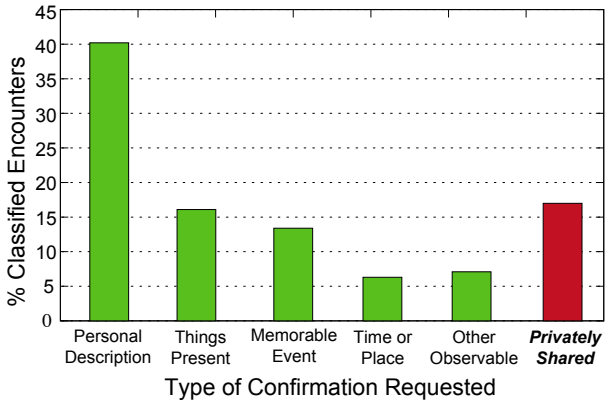

Figure 4: Classification of identity confirmation checks requested, among Craigslist posts requesting some check. Most checks rely on features observable to (and thus forgeable by) third parties, such as a personal description.

Server Synchronization. When convenient (e.g., during nightly phone charging), a user synchronizes encounter and provenance state from the carried mobile device with a *desktop client*. Periodically, on a timescale of hours or longer, the client uploads a preimage-resistant cryptographic hash, $H(x)$, of each new encounter key, x, in randomized order. Randomized order and batched uploads combat server timing attacks on hash uploads. The server might otherwise deduce that a pair of users were involved in an encounter by observing near-simultaneous uploads of the same key hash.

3.2 Missed-Connection Reestablishment

To act upon a missed connection, a user queries the client-side database for the estimated time or place when the encounter occurred (optionally using whatever other provenance information the device and client support). If the client database contains a match, the user may compose a message to be sent to all peers present for the encounter. The client concatenates the message m with a timestamp t, encrypts the result using the encounter key x, and uploads the encrypted token to the server with a hash of the encounter key as $\{H(x); E_x(m||t)\}$.

The server compares the message's key-hash $H(x)$ with all previously uploaded encounter-key hashes. For each match found, the server places the key hash $H(x)$ and the encrypted message token $E_x(m||t)$ in a mailbox for the corresponding user. Clients periodically download new messages and compare the hash associated with each message to their databases of hashes recorded during past encounters. Assuming protocol-compliant client and server behavior, clients will always find at least one match. For each matching key, clients attempt to decrypt the corresponding message from the server. Messages that cannot be decrypted (i.e., because the sender used an unknown encounter key) are discarded.

If a message is successfully decrypted, the client notifies the user, and provides the decrypted message along with any provenance state recorded at the time of the encounter (e.g., time, place, and photographs). The user may choose to respond by entering a reply message m', which the client encrypts and uploads to the server as $\{H(x); E_x(m'||t+1)\}$. Prior to encryption, the client increments the original message's nonce and concatenates it to the reply. The nonce prevents replays of the encrypted message.

Comparison to Common Practice. In the absence of collusion, SMILE guarantees that only encounter participants can decrypt and respond to messages. This significantly reduces the number of potential respondents compared to conventional services in which anyone can respond to a posting. However, a "reconnection" could still be with someone other than the intended recipient if the physical space of the encounter (constrained by wireless transmission range) included other individuals. Once each client has verified the other's proof of co-location, SMILE falls back on the informal checks used in existing missed-connections services. For example, it is common for users to ask for initials, a shirt color, or the subway stop at which an individual departed. Unlike recorded cryptographic state, answers to these questions can be guessed or even forgotten.

Figure 4 categorizes posts on Craigslist requesting some form of verification by confirmation type[1]. Unfortunately, less than 20% of checks could be categorized as "privately shared," while the vast majority of verification information would have been observable to a co-located third party. As a result, though SMILE is more efficient and more secure than existing missed-connections services, it is still vulnerable to fraudulent responses by co-located snoopers.

User Controllable Parameters	
k	Number of users against which encounter anonymity is preserved
l	Number of prefix bits a client reveals from an encounter hash
d	Max random delay, min period to which the server can estimate message timing
f	Fixed number of messages a client sends per conversation
System Properties	
n	Total number of system users
r	Average per-user sending rate to new peers (i.e., rate of new conversations)
p	Proportion of encounters in the system as plausible for a client as encounters in which the client actually participated
c	Proportion of clients users in collusion with the server

Table 1: Analytical Model Parameters

3.3 K-anonymity Preservation

SMILE uses key hashes to deliver messages without compromising users' location privacy, but this does not prevent an adversarial server from inferring which pair of users was involved in an encounter. In this subsection, we discuss the k-anonymity techniques used by SMILE to protect users' encounter privacy and present an analytical model of their properties. The key insight behind these techniques is that by tuning the number of hash bits revealed to the server, l, clients can induce hash collisions to protect their encounter privacy. Furthermore, by controlling the frequency of these hash collisions, users can independently tune their personal level of k-anonymity. Table 1 summarizes our model parameters.

Hash Prefixes. We assume that the central server can associate encrypted messages with a unique source client. Thus, to preserve encounter anonymity, clients obfuscate the message *recipient* rather than the source. Assume clients C_a and C_b participate in an encounter. Let x be an encounter key and $H(x)$ be the corresponding cryptographic hash of the key computed by each device. When up-

[1]The details of our Craigslist classification methodology is described in Section 5.2.

loading encounter hashes to the server, C_a and C_b may not provide all of $H(x)$. Instead, they reveal only $P(H(x), l)$, the l-bit prefix of $H(x)$. C_a and C_b independently choose values for l corresponding to their personally-desired level of k-anonymity and estimated system properties. The ith message m_i in a conversation is uploaded as $\{P(H(x), l); E_x(m_i||t+i)\}$, where x is the encounter key and t is the upload time of m_0.

Matching and Forwarding. The server delivers messages by matching message-hash prefixes to encounter-hash prefixes. Let l_s denote the prefix length of a hash H_s used by the sender of a message. Let l_r denote the prefix length of a hash H_r used by a potential recipient of the message. The server considers $P(H_s, l_s)$ and $P(H_r, l_r)$ a match if and only if one is a valid prefix of the other. That is, $P(H_s, \min(l_s, l_r)) = P(H_r, \min(l_s, l_r))$ for all matching H_s and H_r. Note that for $l_1 \geq l_2$, $P(H, \min(l_1, l_2))$ can be computed as $P(P(H, l_1), l_2)$.

Prefix-Length Selection. As prefix length decreases, the number of messages a client receives will increase. For smaller values of l, a client will receive more messages that do not correspond to any held encounter key. This provides greater anonymity for messages actually intended for that client, at the cost of higher overhead. To limit message flooding, the server may impose a minimum prefix length $l_{\min}$. Clients may similarly filter based on prefix length if the number of received messages becomes burdensome. Users who require $l < l_{\min}$ should abandon SMILE, since the server cannot provide their desired level of anonymity.

Clients must choose their prefix length l carefully. Selection of l should ensure that at least k users send messages using the same key-hash prefix. More precisely, these messages must be sent during a bounded time window d, due to the potential for timing attacks. If l is too large, the server will be able to deduce the identities of two communicating clients from a bidirectional message exchange sharing the same long prefix. In addition, if l is large, the uploaded encounter-key hashes alone may pose a privacy risk, since fewer than k users may even use the same key hash. Because clients will naturally send messages to only a small proportion of peers they encounter, the selection of l to meet messaging requirements necessitates that far more than k users select the same prefix.

To prevent traffic-analysis attacks, the selection of l supersedes the selection of k. There is no privacy risk from a small l, but if l is too small, the client may receive an excessive number of non-decryptable messages. To achieve k-anonymity, the client selects the maximum l that is expected to achieve the desired value of k. Since l is client-tunable, so is k. This is appealing because clients can choose their own level of overhead, depending on their personal level of paranoia.

Anonymity Tuning. The link between l and k is a function of the number n of users in the system, the average rate r at which users send messages to a new peer (i.e., start a conversation or reply for the first time), and the proportion p of encounters in which the receiving client could have participated as plausibly as in its actual encounter. In the absence of any external information linking the receiving client to a subset of encounters, $p = 1$ for the receiver. However, because we assume that the coordinating server has some coarse-grained location information for all clients, p is likely to be less than one in practice.

To strengthen our adversarial model, we assume that the server can predict the precise time t at which a reply message will be sent. If such an attack is thwarted, so are all probabilistic attacks based on reply message timing. Before a reply message is sent, the client introduces a random delay uniformly distributed between 0 and d. During the interval $[t, t+d]$, an average of $n \cdot r \cdot d/2^l$ messages will be sent using the same key hash, $n \cdot p \cdot r \cdot d/2^l$ of which will be indistinguishable from the reply. Assuming uniformly distributed messages, $k = n \cdot p \cdot r \cdot d/2^l$. Thus, a client should select $l = \lfloor \log_2(n \cdot p \cdot r \cdot d/k) \rfloor$ to achieve the desired level of k-anonymity. If $l < 0$, the desired k is unattainable under the system conditions and the client's choice of d.

Parameter Estimation. We expect that a rough estimate of the number of users in the system n and the average new-conversation messaging rate r will be well-known properties of a widely-deployed service. An adversarial server has an incentive to exaggerate these values, convincing a naive client to choose l to be too large for the desired k. Thus, users should rely on external estimates if possible.

An adversarial server may also try to deduce which users are likely to have encountered each other. When effective, this reduces p, the proportion of encounters in the system as plausible as a client's true encounters. We assume that either the server explicitly knows a home physical address for its clients (e.g., as part of billing information) or can deduce a localized geographic region from a client's IP address during message and key hash uploads. There is a high probability that encounters occur in the geographic region surrounding a client's home. An adversarial server can thus correlate matching encounter-key hashes coming from the same geographic location. Thus, p is limited to the proportion of recorded encounters in the plausible encounter region defined by a client's address. The size of this region is dependent on the mobility patterns of both the client and the peers the client encounters.

If a client can accurately approximate p, l can be correspondingly deflated to preserve the desired k, albeit at increased overhead. Unfortunately, estimating p is difficult. One reasonable heuristic would be to compute the average maximal distance a user travels from his home between synchronizations, assume this distance to be the radius of a circular encounter region, multiply by the population density, multiply by the proportion of people in the area who use the service, and divide by the number of users in the system. Since such techniques may impose high user burden, users may simply select a widely-accepted conservative value and request that the server aggressively filter messages.

Filters. Lower values of p lead to greater user overhead in the number of non-decryptable messages that must be received to preserve k-anonymity. To combat this, the user may specify that she only wishes to receive messages from users residing within the same geographic region. Moreover, the user may request that the server filter messages on the basis of any number of other attributes the server knows about the sender. This is especially appealing for missed-connections services. For example, a user may request to only receive messages from people of the opposite sex and within a certain age range. This is similar to the way users search for peers on popular dating websites.

Collusion Attacks. Although SMILE provides protection against channel snooping by the server or individual clients we cannot defend against all client-server collusion attacks. If clients present for the encounter collude with the server, they can provide the server with the encounter key used for end-to-end encryption. Clients can also aid the server in deanonymizing attacks. To precisely determine a client's identity, the server would need to collude with $k-1$ of the client's k randomly-selected anonymizing peers. Although this is unlikely, partial collusion weakens the anonymizing properties of the system. To compensate, if a user anticipates that the proportion of peers in collusion with the server is c, it should actually estimate $k = n \cdot p \cdot (1-c) \cdot r \cdot d/2^l$, and thus select l as follows:

$$l = \left\lfloor \log_2 \left(\frac{n \cdot p \cdot (1-c) \cdot r \cdot d}{k} \right) \right\rfloor \quad (1)$$

Attacks on Nonuniform Distribution. Note that the model we have presented thus far assumes that the distribution of messages is uniform across clients. In practice, we expect that some pairs of communicating clients will deviate from the average conversation length. Furthermore, we expect that conversations will be synchronous. If a client C_a sends a message to C_b, C_a generally will not send another message to C_b until C_b replies. This implies that a pair of communicating clients will send approximately equal numbers of messages to the same hash prefix, making the pair easier to identify.

To combat these attacks on nonuniform message distribution, SMILE requires that all conversations be of a fixed length f. Whenever a client chooses to send a new message, or replies to a message for the first time, the client commits to sending precisely f messages. All clients may choose f independently, but they must not select f to be conversation dependent. No more than f messages may be sent for the same encounter. If a conversation ends before f messages are sent, the client pads the conversation with dummy messages. A dummy message for key x is constructed as, $\{P(H(x), l); E_y(m_r)\}$, where m_r is a random message and y is a random key.

To prevent intra-conversation timing attacks, messages must be sent at regular intervals. When a conversation is initiated, f upload times are predetermined. A new message is not uploaded until the next upload time. If the user does not create a message before some upload time, a dummy message is sent instead. Limiting users to a bounded f messages per encounter should not substantially limit functionality within the missed-connections domain. Within a few messages, it is likely that users would be willing to forgo some mutual anonymity to negotiate an out-of-band channel, such as email or telephone.

Alternate Adversarial Model. These k-anonymity techniques are designed to protect against an adversarial server which employs active attacks to aggressively deanonymize user encounters. We may also consider a weaker adversarial model, where the server is not malicious, but is vulnerable to read/write attacks on its database. Such a server may provide accurate estimates of n and r directly to its users and aid in determining p. It would not collude, thereby setting $c = 0$. Clients may compute the appropriate l trivially for any desired k. Note that the choice of random message delay d is still an important input to the selection of l from k, as it affects the anonymity of records stored in the database. Similarly, users must still account for attacks on nonuniform message distribution by using a fixed conversation length f.

Messaging Overhead. For a given set of system parameters, we can compute the expected overhead of received messages. (2) provides the number of recipients for any message. Assuming that users optimally select l as in (1), the number of recipients does not depend on n. Discretized l selection ($l = \lfloor l^* \rfloor$, l^* optimal) is accounted as $z \in [1, 2)$.

$$\text{Number of recipients} = \frac{n}{2^l} = \frac{z \cdot k}{p \cdot (1-c) \cdot r \cdot d} \tag{2}$$

Multiplying by the average rate r at which users send messages to new peers, we see that the rate at which messages are received does not depend on the rate at which messages are injected into the system. (3) presents a formula for computing the ratio of the number of clients that receive a message as overhead to the total number of clients receiving the message. This is an upper bound, which assumes that all messages are intended for a single recipient. For encounters with multiple co-located peers, the number of clients for which the message is decryptable will increase (we do not consider such messages overhead).

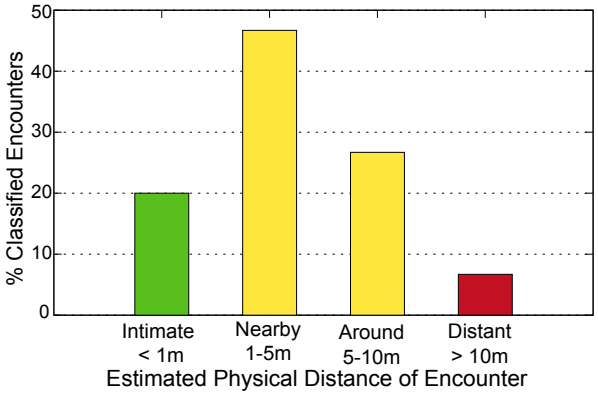

Figure 5: Estimated Craigslist encounter distance. Only ≈5% of encounters occur outside of Bluetooth range.

$$\begin{matrix}\text{Overhead} \\ \text{proportion}\end{matrix} = \frac{n/2^l - 1}{n/2^l} = 1 - \frac{p \cdot (1-c) \cdot r \cdot d}{z \cdot k} \tag{3}$$

In (4), we compute the rate of overhead messages received by a client. Under the assumption of one recipient per message, the average rate at which a user receives decryptable messages is equal to the average per-user sending rate r.

$$\text{Overhead reception rate} = \frac{z \cdot k}{p \cdot (1-c) \cdot d} - r \tag{4}$$

These results show that message overhead depends on z, k, p, c, and r, but is not directly affected by increases in the total number of system users n. As the rate r at which legitimate messages are injected into the system increases, both the proportion of messages received as overhead and absolute rate of received overhead messages decreases. Appealingly, efficiency increases with usage.

3.4 Implementation Considerations

Sensing Platform and Key Distribution. We envision SMILE running on mobile phones and laptops. Thus, compatibility with currently-deployed technology is an important consideration. Fortunately, readily-available wireless communication platforms can be used for key distribution. Bluetooth, given its appropriate range (≈10m) and low power consumption, is an obvious choice. In Figure 5, we note that the effective range of Bluetooth is comparable to the observed distance in the vast majority of Craigslist posts. Note that there is a security versus performance tradeoff in wireless transmission range. Shorter ranges limit potential snooping attacks while higher ranges increase the probability of encounter detection.

Other low-power radio platforms such as IEEE 802.15.4 ZigBee may be appropriate, but we only considered solutions that are readily-available on commodity hardware. Given the tradeoffs, we believe that Bluetooth provides the most reasonable platform in widespread use. We have implemented an encounter detection and key sharing scheme based on Bluetooth discoverable-mode service advertisements. The use of service advertisements obviates the need for device pairing and avoids breaking compatibility with other concurrently-running Bluetooth services.

WiFi (802.11) is another widely-available option. WiFi's relatively larger range, and increased ability to penetrate walls, provides a poorer approximation of the type of co-location guarantee that Craigslist users desire from a missed-connections service. While transmission power control may effectively limit range to

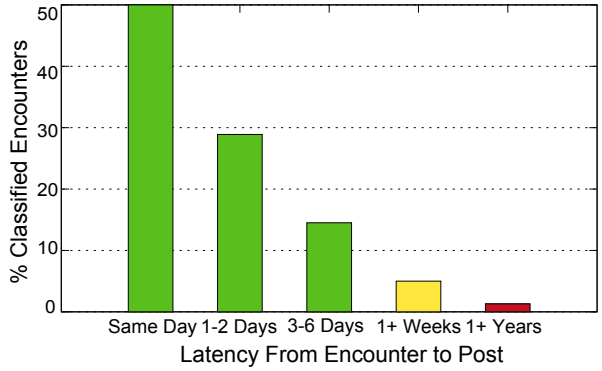

Figure 6: Estimated latency from time of encounter occurrence to Craigslist post.

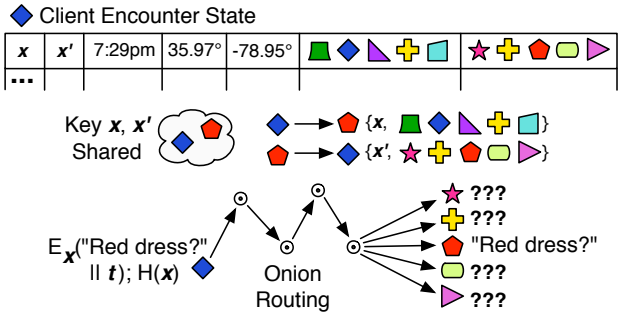

Figure 7: Distributed scheme operation. During an encounter, each peer shares k identifiers and an encounter key. Messages are sent using onion routing or an anonymous remailer to preserve anonymity.

the desired distance, there are other practical problems. For example, key distribution would require all devices to share the same channel. The problem becomes simpler if we assume that nearby participants associate to the same WiFi access point, in which case broadcast packets can be used trivially. Alternatively, WiFi could be switched to ad-hoc mode, with keys broadcast as the network SSID. The primary drawbacks of this approach are the high power draw of 802.11-beacon scanning and the loss of Internet connectivity over WiFi.

Advanced techniques exist for establishing a shared key over a wireless channel. Radio-telepathy [24] and [13] exploit the symmetry of time-varying wireless-channel fading to establish a shared key between a transmitter and receiver, and are invulnerable to snooping by other local adversaries. While such techniques could be used in SMILE for key distribution, they are unnecessary for our attacker model. SMILE relies on cryptography only to prove that an encounter occurred with *any* co-located peer. All local peers are equally legitimate. In the case that multiple peers are encountered simultaneously, messages sent using the corresponding key should be readable by all.

Storage Burden. Using Bluetooth as a key-distribution mechanism prevents SMILE from detecting extremely brief encounters because of its relatively slow service-scan speed (≈30s). As a result, clients will record at most two entries per minute, per encounter of the form shown in Figure 1. This provides an upper bound on the number of encounter keys recorded since packet loss will reduce the actual number. From these bounds, we define an *encounter-time* metric to measure the duration of a continual encounter with a single peer that can be supported by some quantity of storage.

Using a 128-bit encounter key, a 32-bit pre-computed encounter-key hash as a computational optimization, a 32-bit timestamp, and a 32-bit latitude and longitude, each entry has a fixed 256-bit (32-byte) storage requirement. Assuming no additional local state is maintained, a client can maintain 524,288 encounter entries (182 encounter-days) using a conservative 16 MB of disk storage. From Figure 6, we see that, at the extreme end, missed-connections attempts rarely occur beyond a period of one year. If we discard encounter data after one year, 16 MB is enough to sustain an average of 1,436 encounter entries (12 encounter-hours) per day.

The server maintains a 32-bit client id, 32-bit hash prefix, and 8-bit hash prefix length, for a total of 72 bits per encounter entry. Since each encounter is recorded by both clients, 144 bits are used per unique encounter entry. The server should organize its database of new encounter entries in FIFO order. When storage limits are reached, the server should evict the oldest entries. 18 GB is enough server storage for a billion encounter entries, or more than one encounter-millennium. 500 GB provides enough storage for $\approx$ 100 encounter entries for every person in the US. From these calculations, we do not expect storage overhead costs to limit system adoption.

4. DECENTRALIZED ARCHITECTURE

In this section, we consider a decentralized scheme as an alternative to our centralized design. It provides similar privacy guarantees without requiring a dedicated server, but yields different tradeoffs. The primary drawback is that it requires the use of an anonymizing network to provide an untraceable messaging service between participants. For this purpose, an anonymous remailer (e.g., Mixmaster [27] or Mixminion [26]) or general onion routing (e.g., Tor [5]) may be used. Note that this work is also complementary to more advanced anonymized messaging techniques, such as information slicing [17]. The anonymized messaging channel protects peer identities from a malicious third party. Our k-anonymizing techniques extend this protection to allow bidirectional communication without revealing identities to peers engaged in the exchange.

We consider our server-based design more practical in terms of deployability and messaging overhead, but present this alternative to provide a more complete perspective on the design space. The distributed approach extends work we presented previously [22]. From our prior design, we have removed the need for a coordinating server, which eliminates a number of location privacy vulnerabilities at the cost of higher per-encounter wireless transmission requirements. Messaging overheads are equivalent. Rather of focusing on these improvements, this discussion highlights differences from our centralized scheme. Where details are omitted, the techniques are similar to those used in SMILE. Figure 7 illustrates the decentralized scheme.

4.1 Distributed Operation

Each participant chooses a unique personal identifier to enable peer-to-peer communication (e.g., an email address, instant-messaging screen name, domain name, or IP address). To maintain their anonymity, users should choose an identifier that cannot be mapped to their actual identity. Users must also use an anonymizing network for all communication. To simplify our discussion, we assume that users choose email-address identifiers, and that all communication is handled by an anonymous remailer, such as Mixmaster.

Anonymity. In addition to a personal identifier, users maintain another set of identifiers, any of which could plausibly be under their control. Users preserve their k-anonymity by sending messages through an anonymizing mix network with the source specified as a tuple of plausible identifiers called an *identifier set*. A tuple of size k only reveals that one or more of the k identifiers was present at the encounter. Encounter privacy is a function of the plausibility that a message could have come from any member of the tuple with equal probability.

Anonymous Messaging. As in the centralized case, our messaging scheme provides a channel that is end-to-end confidential and resistant to man-in-the-middle attacks. During an encounter at time t, peers use wireless transmissions to exchange messages of the form $\{I, x\}$ where $I = \{i_1, i_2, \ldots, i_k\}$ is the source peer's identifier set and x is an encounter key. Later, assume peer P_a with $I_a = \{a_1, a_2, \ldots, a_k\}$ wishes to send a message m to previously-encountered peer P_b with $I_b = \{b_1, b_2, \ldots, b_k\}$. P_a sends an email containing $\{H(x); E_x(I_a||t||m)\}$ through an anonymous remailer to all identifiers in I_b. P_b may reply with $\{H(x); E_x(I_b||t+1||m')\}$ to all $a_i \in I_a$. Note the use of an incremented timestamp nonce prevents a variety of replay attacks. Receiving peers not actually present for the incident of encounter will not be able to decrypt the message or the contained identifier set, thereby gaining no information.

4.2 Identifier Set Selection

The privacy properties of the decentralized scheme are tied to the quality of a user's identifier set. Poor selections may allow adversaries to establish a mapping between an identifier and the source of an anonymized message, and, as a result, reveal the identify of a previously-encountered peer.

Identifier Collection. Before an identifier set is selected, users should collect at least k other identifiers. A user's device can build a database of identifiers by recording the identifier sets of other participants. Collecting identifier sets this way is appealing because it promotes identifier dispersion: the more widespread a user's identifier is, the more likely it is to be used in other users' identifier sets. Broad re-use of an identifier makes the corresponding user's actual locations more difficult to infer due to the increased number of false positives.

Collusion Attacks. As in the centralized scheme, collusion can weaken anonymity guarantees. For each of a message sender's $k-1$ anonymizing identifiers in collusion with the message recipient, k is effectively decreased by one. Given that an adversarial peer (encountered or otherwise) has no way to control the contents of another user's identifier set, this would be a difficult attack.

Geographic Plausibility. Given that a message recipient will have some knowledge of a sender's general whereabouts (e.g., from the time and place of the encounter itself), it may be possible to pinpoint out-of-place identifiers. To increase the plausibility of peer identifiers used in the set, it is desirable to maintain some general locality with the encounter. This may be accomplished by discarding known identifiers after an expiration period, assuming that recently-acquired identifiers will tend to correspond to users located in the general vicinity.

Bootstrapping. The maximum size of an identifier set is limited by the number of peers of which a client is aware. In cases where an insufficient number is known, it may be helpful to introduce additional false identifiers. If an adversarial peer is likely to have trouble distinguishing legitimate from contrived identifiers, this may provide additional protection. Unfortunately, not all false identifiers are equally plausible. For example, we assume email address used directly as identifiers. Plausible fake email addresses should be human readable (which would be feasible through dictionary-based random generation) and not cause mail to bounce.

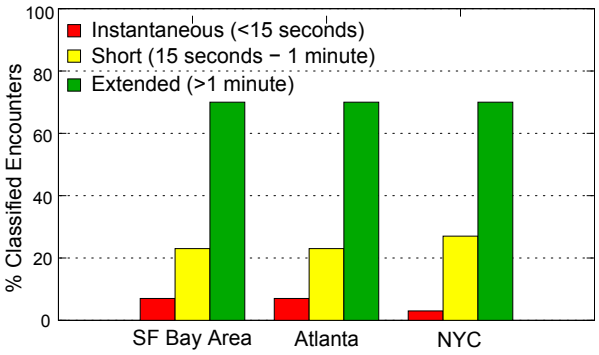

Figure 8: Estimated encounter duration implied by Craigslist posts, by geographic locale.

Slow Evolution. Once a user has collected enough plausible identifiers, she must select an appropriate subset for use during encounters. First, use of newly-added identifiers should be delayed, so that they are not reused in a time-linkable reencounter with their source. Moreover, a user's identifier set should change slowly over time to prevent an adversary from linking multiple encounters. To see why, assume P_a shares I_1 at time t_1 and I_2 at time t_2. P_b encounters P_a at t_1 and P_c at t_2 and able to guess that $P_c = P_a$ through external information. For example, if P_a is the only other person around at both times, the link is clear. Now, for both t_1 and t_2, P_b can deduce that the true identifier of P_a is in $I_1 \cap I_2$, or more generally, $\cap_{i=1}^{n} I_i$ for n encounters. This attack is most difficult to prevent for adversaries encountered on a regular basis. However, such a well-positioned adversary can likely deduce personal information more easily from other means.

Identifier-Set Size. Users must carefully select an appropriate identifier set size s. Clearly, s must be greater than k to preserve k-anonymity. In practice, due to uncertainty over the above identifier inclusion criteria, participants should select $s \geq k/(1-j)$, where j represents the probability that an adversarial peer can reject an included identifier as less plausible than the user's true identifier. Practical considerations, such as reasonable message length or an insufficient number of known peers, may bound s below the ideal selection. As in the centralized scheme, there is a privacy-versus-overhead tradeoff in choosing an identifier-set size.

5. EVALUATION

In this section, we consider deployment feasibility for SMILE, including (1) the ability of our passive key-exchange protocol to adequately establish shared-key state among encountered clients, and (2) the appropriateness of our system properties for real-world missed-connections usage.

5.1 Key Advertisement Detection

The feasibility of SMILE depends on its ability to reliably detect encounters within a potentially short amount of co-location time, since co-location proofs can only be provided with shared-key state. The shorter this minimum duration is, the more widely applicable our scheme will be.

Target Applications. Long-duration detection is sufficient for activities such as shared meals, a conversation over coffee, or mutual attendance at a seminar. Detecting short events, such as a quick hallway passing, requires a faster exchange. Romantic queries, business propositions, or friend-seeking searches may be the result

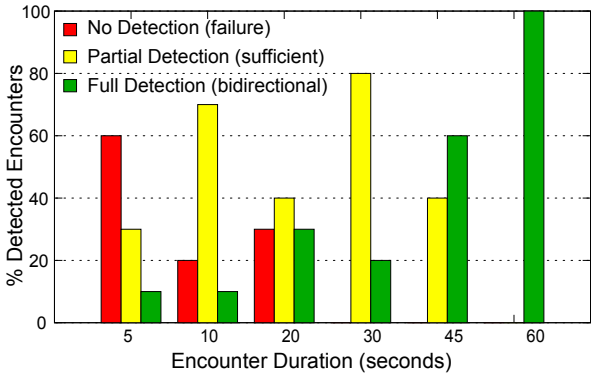

Figure 9: Encounter-key discovery. Each detection scan begins 15 seconds after the completion of the prior scan.

of encounters that are only tens of seconds long. Figure 8 shows that less than 10% of the Craigslist encounters in our study lasted 15 seconds or less.

Bluetooth Detection Test. In our implementation, client devices periodically initiate available service scans for all Bluetooth devices in range. Service names identified as keys are recorded in a local relational database, as are all self-advertised keys, along with the current time and coordinates, as determined by Skyhook WiFi-based localization. After completing a scan, the client pauses, chooses a new key to advertise, and then initiates the next scan. In Figure 9, we show reliability results for co-location detection and key exchange. In these experiments one client remained stationary in a room while the other client started out of range, walked into the room, remained stationary for the specified interval, and then exited.

We categorized detection as mutual (when both keys received by both clients), partial (when one key received by one client), or failed (when neither client received a key). Our protocol only requires partial detection, since the failing client will have recorded the broadcast key and successfully shared it with the other client. Our results show that Bluetooth key advertisement and scanning can reliably detect encounters at timescales of 30-60 seconds, which is acceptable for the vast majority of encounters we found on Craigslist (Figure 8). For this test, we selected a pause period of 15 seconds from the end of one scan to the start of the next. Given the speed of detection, it may be preferable to extend this interval for a corresponding reduction in energy consumption, while still meeting application-appropriate detection speed requirements.

5.2 Craigslist Classification

To ground our assumptions of how people might use SMILE, we examined hundreds of Craigslist posts from US-metro areas and classified the described encounters by identity-confirmation checks (Figure 4), distance between the to-be-reconnected individuals (Figure 5), the time between encounter and posting (Figure 6), and encounter duration (Figure 8). Each post's content was manually classified.

We examined Craigslist missed-connections posts from a number of US-metro areas including Chicago, New York, Philadelphia, Raleigh, San Diego, Seattle, the San Francisco Bay Area, and Washington DC. We ignored sub-classification by geographic district or gender-based filters. Posts were selected systematically: beginning with the most recent posts, we examined each post in reverse chronological order until at least 100 legitimate posts had been classified. By its open, bulletin-board nature, Craigslist is prone to some misuse and abuse, including spam, incendiary rants, and cryptic language. Ambiguous or indecipherable posts, posts that did not represent a missed connection, and posts with obscene content were not considered. All Craigslist figures in this paper present a histogram of the proportion of legitimate posts within each category. The scope of our Craigslist study was limited and our methodology was not scientifically rigorous, but the data we collected provides valuable initial insight into the challenges that emerging systems such as SMILE would face if deployed.

6. RELATED WORK

SMILE lies at the intersection of three research areas: location proofs, location privacy, and anonymized communication.

Location Proofs. Several systems have recently sought to give end users the ability to prove that they were in a particular place at a particular time. [30] proposed a solution that is suitable for third-party attestation, but relies on a PKI and changes to the 802.11 access-point infrastructure. SMILE takes a more ad-hoc approach and requires no changes to existing infrastructure, but generates proofs that only demonstrate a mutual encounter.

In [18], the authors describe a secure localization service that can be used to generate unforgeable geotags for mobile content such as photos and video. The primary difference between this work and ours is that it relies on the wide deployment of secure infrastructure to generate proofs, while we rely on users to prove that an encounter occurred.

SPATE [19] is similar to SMILE in its ad-hoc design and also uses physical encounters to allow users to establish private communication channels. SPATE was designed with the assumption that its users already know each other, and at the time of their physical encounter intend to communicate sometime in the future. Our design assumes its users do *not* know each other, and provides a mechanism to communicate in retrospect of an encounter, while maintaining anonymity.

In a preliminary version of this work [22], we targeted missed-connections and utilized similar wireless techniques to prove when an encounter occurred. However, this service was prone to linking attacks by malicious servers since users reveal their actual location information to the service provider. SMILE avoids such attacks by forwarding key hashes to the server instead.

Location Privacy. A number of projects have investigated the use of trusted central servers to anonymize location information, especially to meet k-anonymity requirements [9, 16]. Our approach provides similar k-anonymity guarantees, but without requiring that the third-party service be trustworthy. Other work attempts to provide location privacy though access-control mechanisms [12] and digital rights management [11]. Both models rely on a trusted server to manage users' location information.

Adeona [29] is a device-tracking service designed to help users recover lost or stolen mobile devices without compromising their location privacy. Like SMILE, Adeona uses pseudo-random generators to name and encrypt users' location information. The key difference between the two services is the way that they compute location identifiers. The location identifiers generated by an Adeona-enabled device only need to be meaningful to the individual device owner since only she is allowed to track her device. On the other hand, SMILE must allow independent, co-located users to deterministically compute the same encounter identifier without revealing any information about the encounter's place or time to entities that were not present.

Finally, SmokeScreen [4] is a mobile social service that uses short-range wireless messages among co-located users to en-

able "presence-sharing." Smokescreen is primarily meant to enable privacy-preserving presence-sharing among users with pre-established trust relationships, and relies on centralized, trusted brokers to coordinate anonymous communication between strangers. SMILE allows co-located strangers to communicate without revealing their location or mutual interest to service providers.

Anonymous Messaging. Anonymous remailers [27, 26] and general onion routing [5] provide a communication channel that is anonymous to third-party adversaries. Information slicing [17] provides similar guarantees, but without need for public-key cryptography. Our techniques provide an additional level of privacy, where even those individuals participating in a message exchange maintain mutual k-anonymity.

7. CONCLUSION

This paper has described the SMILE mobile social service. SMILE aims to provide an efficient missed-connections service using mobile devices without relying on trusted coordinating servers or pre-established trust among users. To meet this goal, SMILE relies on trust derived from physical encounters among users. Co-located SMILE devices establish trust with each other by performing a passive key-exchange protocol that can be used to generate a proof of their encounter. Clients only need to share hashes of their logged encounter keys with the SMILE server since hashes protect users' location and encounter privacy from malicious servers and peers. Through protocol analysis, study of the Craigslist missed-connections service, and experimentation with a prototype SMILE implementation, we demonstrated the strong privacy guarantees and feasibility of SMILE.

8. ACKNOWLEDGEMENTS

We sincerely thank our anonymous reviewers for their invaluable feedback. We thank Hakan Seyalioglu (UCLA) and Peter Gilbert (Duke) for comments on preliminary drafts. Landon Cox was partially supported by NSF CAREER Award 0747283.

9. REFERENCES

[1] G. D. Abowd, G. R. Hayes, G. Iachello, J. A. Kientz, S. N. Patel, M. M. Stevens, and K. N. Truong. Prototypes and paratypes: Designing mobile and ubiquitous computing applications. In *PerCom*, 2005.

[2] Alexa. Alexa the web information company. `http://www.alexa.com/`.

[3] S. Consolvo, P. Klasnja, D. W. McDonald, D. Avrahami, J. Froehlich, L. LeGrand, R. Libby, K. Mosher, and J. A. Landay. Flowers or a robot army?: encouraging awareness & activity with personal, mobile displays. In *Ubicomp*, 2008.

[4] L. P. Cox, A. Dalton, and V. Marupadi. Smokescreen: flexible privacy controls for presence-sharing. In *MobiSys*, 2007.

[5] R. Dingledine, N. Mathewson, and P. Syverson. Tor: the second-generation onion router. In *USENIX Security*, 2004.

[6] N. Eagle and A. Pentland. Social serendipity: Mobilizing social software. *IEEE Pervasive Computing*, 4(2):28–34, 2005.

[7] Google Mobile. Latitude. `http://www.google.com/latitude/`.

[8] B. Greenstein, D. McCoy, J. Pang, T. Kohno, S. Seshan, and D. Wetherall. Improving wireless privacy with an identifier-free link layer protocol. In *MobiSys*, 2008.

[9] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *MobiSys*, 2003.

[10] M. Gruteser and D. Grunwald. Enhancing location privacy in wireless lan through disposable interface identifiers: A quantitative analysis. *ACM MONET*, 2005.

[11] C. Gunter, M. May, and S. Stubblebine. A formal privacy system and its application to location based services. In *Privacy Enhancing Technologies (PET)*, 2004.

[12] C. Hauser and M. Kabatnik. Towards Privacy Support in a Global Location Service. In *Proc. of the IFIP Workshop on IP and ATM Traffic Management*, 2001.

[13] S. Jana, S. N. Premnath, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy. On the effectiveness of secret key extraction from wireless signal strength in real environments. In *Mobicom*, 2009.

[14] T. Jiang, H. J. Wang, and Y.-C. Hu. Preserving location privacy in wireless lans. In *MobiSys*, June 2007.

[15] John Leyden. Teen hack suspects charged over myspace extortion bid, May 2006. The Register.

[16] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing Location-Based Identity Inference in Anonymous Spatial Queries. *IEEE Trans. KDE*, 2007.

[17] S. Katti, J. Cohen, and D. Katabi. Information slicing: Anonymity using unreliable overlays. In *NSDI*, April 2007.

[18] V. Lenders, E. Koukoumidis, P. Zhang, and M. Martonosi. Location-based trust for mobile user-generated content: applications, challenges and implementations. In *HotMobile*, February 2008.

[19] Y.-H. Lin, A. Studer, H.-C. Hsiao, J. M. McCune, K.-H. Wang, M. Krohn, P.-L. Lin, A. Perrig, H.-M. Sun, and B.-Y. Yang. Spate: Small-group pki-less authenticated trust establishment. In *MobiSys*, 2009.

[20] Loopt, Inc. Your social compass | loopt. `http://www.loopt.com`.

[21] Louise Story and Brad Stone. Facebook retreats on online tracking, November 2007. The New York Times.

[22] J. Manweiler, R. Scudellari, Z. Cancio, and L. P. Cox. We saw each other on the subway: secure, anonymous proximity-based missed connections. In *HotMobile*, 2009.

[23] D. W. Margo and H. U. Margo Seltzer. The case for browser provenance. In *TaPP*, 2009.

[24] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *Mobicom*, 2008.

[25] Megan McCarthy. How Facebook employees break into your profile, November 2007. `http://www.valleywag.com`.

[26] MixMinion. Mixminion anonymous remailer. http://www.mixminion.net.

[27] U. Moller, L. Cottrell, P. Palfrader, and L. Sassaman. Mixmaster protocol — version 2. *IETF Internet Draft*, 2003.

[28] K. Muniswamy-Reddy, D. Holland, U. Braun, and M. Seltzer. Provenance-aware storage systems. In *USENIX*, 2006.

[29] T. Ristenpart, G. Maganis, A. Krishnamurthy, and T. Kohno. Privacy-Preserving Location Tracking of Lost or Stolen Devices: Cryptographic Techniques and Replacing Trusted Third Parties with DHTs. In *USENIX Security*, 2008.

[30] S. Saroiu and A. Wolman. Enabling new mobile applications with location proofs. In *HotMobile*, 2009.