RESEARCH ARTICLE OPEN ACCESS

Requirement and Design of Secure Encounter-based Mobile Social Networks

Kilaparthy.Punyavathi Pushpa [1], Panthangi.Pavithra [2]
Dept. of CSE, Visakha Institute of Engineering and Technology,
Andhra Pradesh, India

ABSTRACT

Encounter-based social networks and encounter-based sys- tems link users who share a location at the same time, as opposed to the traditional social network paradigm of linking users who have an offline friendship. This new approach presents challenges that are fundamentally different from those taken in charge by previous social networking functions. In this report, we explore the operational and security require- months for these new systems, such as availability, protection, and privacy, and present various design alternatives for building secure encounter-based social nets. To highlight these challenges we examine one recently proposed encounter-based social network design and compare it to a set of idealized security and functionality demands. We demonstrate that it is vulnerable to several attacks, including impersonation, collusion, and privacy breaching, even though it was designed specifically for security. Aware of the possible pitfalls, we construct a flexible frame- work for secure encounter-based social nets, which can be utilized to build networks that offer different security, secrecy, and availability guarantees. We describe two example constructions derived from this framework, and consider each in terms of the ideal demands. Some of our new designs fulfill more requirements in terms of arrangement protection, dependability, and privacy than previous work. We also evaluate real-world execution of unitary of our designs by putting through a proof-of-concept iPhone application called MeetUp. Experiments highlight the potential of our system and hint at the deployability of our designs on a great plate.

Keywords:- **Social networks, Location-based services.**

I. INTRODUCTION

In the formal model of social networks, users select their contacts from a set of off-line acquaintances. Despite their usefulness, these conventional networks support only a subset of social networking: two users will merely be able to build a kinship in the social network if they know of, or are introduced to each other. On the other hand, in an encounter-based social network, the sole prerequisite for establishing the connection is to be in the same space at the same time similar to hitting up a conversation in a public space. Brush-based social networks would offer a computing infrastructure to grant for creation of varied services such as a "missed connections" virtual bulletin board, on-the-fly introductions (business card exchange), or real-time in-person key distribution to bootstrap secure communication in other organizations.

Although at first glance encounter-based systems seem very similar to existing social nets, they present a dra- matically different set of challenges, not the least of which are security and privacy of users and authenticity of the other party in a conversation. Warrants that are little in traditional social networks, such as authenticity (ensuring one is coming-municating with the desired person), become open problems in encounter-based webs. Additionally, requirements like anonymity a feature that is not required in most traditional online social networks based on prior face-to-face contact need to be considered in encounter-based webs. This is desirable because users would expect information about people they bump to meet to remain secret. Furthermore, since people do not automatically set their faith in others simply based on presence in the same location, it is also suitable to disclose the minimum sum of data asked for future secure communication. Sharing detailed personal information is not the primary destination of the encounter-based networks, but can of course be easily gone through if both users agree upon the successful verified encounter.

In this report we consider fundamental requirements for encounter-based social nets. We notice that in addition to basic functionality like high availability, scalability, and robustness to failure, these organizations should offer several security guarantees, including privacy in the form of unlink-ability of users sharing an encounter, confidentiality of data exchanged among encounter participants, and authentication of both users in a two-party conversation. We show that SMILE [27], a recent state-of-the-art design, fails to conform to a number of these necessities (even though it was made explicitly with security in mind). We offer a generic plan that can be applied to build networks that offer different security guarantees. We then describe individual designs and show the benefits and trade-offs of specific security design decisions.

Unlike prior work, we provide fine-grained separation be-tween the encounter event and the eventual connection and communication: authentication and communication may hap-pen immediately, or may be delayed for an arbitrary period of fourth dimension. The former provides unlinkability between the two paired users (a third party cannot determine that two users have established a connection), while the latter increases convenience and flexibility at the price of somewhat

degraded unlinkability. Nonetheless, both schemes guarantee authentication that once installed, the connector is desired with the user. Both of these plans consist of an "online phase," where the meeting takes place and encounter instance information is exchanged, and an "offline" or delayed communication phase, where encounter information is used for the two parties to reconnect and communicate privately. It is worth mentioning that we accept that other users at the encounter time and locale- tion are potentially malicious, and may collect information, collusion with other parties, and otherwise make it difficult for two people to set up a secure private link. We got a prototype of our invention, called MeetUp1, that uses visual authentication for encounter information exchange and verification. At the heart of our system is a visual au- authentication scheme that provides authenticity guarantees for users affected in an encounter. Our authentication scheme capitalizes on that people are good at remembering faces, but bad at thinking of names. Brush-based networks with visual authentication would work to people's strengths, letting anyone who calls up a face to later connect with the "owner" of that face, without the need to call up additional information. MeetUp uses Tor hidden services [14] to provide an anonymous communication channel for the second stage of our protocol. By performing preliminary real-world experiments using plausible deployment settings, and considering user feedback, we highlight the end-user usability of our organization and its feasibility for deployment at larger scales. While the primary contribution of this report is an encounter- based social web design, our techniques can be used for a full scope of applications, such as a drop-in substitution for a face-to-face key distribution service for future secure communication, e.g. SPATE [23], or for privacy-preserving file sharing systems, e.g. OneSwarm [20]. In OneSwarm, untrusted users get their keys from an online key distribution center. Using our design, one may distribute keys to interested users based on some shared activity—an encounter. Any application that takes a central pre - distribution, such as warehousing services, private file-sharing arrangements, private collaboration groups, etc., would benefit from our conception in the same manner. Some other ex-ample is a scientific meeting, where some researchers present their study, and others take part in discussions, and no one has time to introduce themselves to everyone. We can employ our encounter-based arrangement for private on-the-fly name and business card distribution—concrete models are discussed In §6. 4.

Our contributions in this work are as follows. (i) By first outlining security and operational demands that are unused-

Ally desired to encounter-based social network and arguing that these are minimal prerequisites for many distributed systems with reasonable security and privacy guarantees, we analyze the extent to which SMILE, a recent state-of-the- art design of secure encounter-based social network, meets these requirements, proving that it is vulnerable to many approaches. (ii) We propose a new and generic architecture for encounter-based social networking that greatly differs from the architecture of previously proposed schemes and indicate two possible implementations, each striking a balance between performance and protection. (iii) We show the feasibility of our designs by putting through a proof-of-concept system—including an iPhone application called MeetUp—conforming to our requirements and assessing its performance in real-world settings using mobile devices, and by bringing further evidence on the serviceability of our design and rationality of use assumptions based on several user surveys.

The arrangement of this oeuvre is as follows. In §2 we de-scribe idealized security and operational requirements expected to encounter-based webs. In §3 we discuss some of the related work in the literature, observed by a discussion of vulnerabilities of SMILE. In §4 we introduce the intention of a generic encounter-based social network and discuss two spas- cific designs. In §5 we discuss the implementation of MeetUp, and details of some of the experiments that we performed to illustrate the usability of our intent. In §6 we highlight the main discussion points, followed by concluding remarks in §7.

II. REQUIREMENTS AND CHALLENGES

Equally we have mentioned in §1, many encounter-based designs do not take even basic security and privacy requirements along with functionality and public presentation. Others fail to see these requirements even though they were made with the expressed goal of living up to them. Beneath, we explore some requirements for idealized secure encounter-based social net- kit and boodle. While this list is by no means perfect, it can be applied as a preliminary template for evaluating past and future plans.

2.1 Security Requirements

Here we outline some of the desired security features of encounter-based social nets. Remark that these requirements are generic in the sense that they may apply to many dis-tributed systems which combine human interaction, sensitive private information, and network communication. The security demands we expect in these arrangements are as follows.

(i) privacy or unlinkability. The seclusion of two parties sharing an encounter must be protected, even from others in the neighborhood who may likewise take part in simultaneous meetings. In this case, privacy means that an external adversary (even one taking part in the encounter or colluding with a "bulletin board" or rendezvous server to be used in latter phase) who is not one of the two users of interest should not be able to conclusively find that two users have established a link.

(ii) authenticity, meaning that when two users decide to make a connection, they should be assured that messages indeed

2.2 Functional Requirements

The following are generic functional requirements in the con-text of large-scale distributed systems that are also suitable for

an encounter-based social net. (i) availability. As such, the infrastructure to exchange meeting information should be accessible most of the time. The unavailability of individual users should not regard the accessibility of other users. Since the time at which encounter parties check for potential encounters associated with their natural processes could be arbitrary, the encounter-based social net is more sensitive to availability than conventional social networks. (ii) scal- ability. With typical social networks being large in size, any potential social network design, including those based on encounters, should scale to back up a large act of concurrent users. This requires minimizing dependence on a centralized entity (our rendezvous server mentioned above).

III. BACKGROUND AND RELATED WORK

While it may seem that enforcing these requirements would be straight, it is surprisingly challenging in practice. Recently, Manweiler et al. Devised SMILE [27] to implement a subset of these demands. While they succeed in taking on some of the operational requirements, their system does not protect against a number of common security exposures, such as the "man-in-the-heart" (or MitM) attack, which passes to several other breaches as shown under. Nearly connected to our workplace, as comfortably as to SMILE, are GAnGS [9] and SPATE [23], which are both systems built to facilitate secure information interchange among groups in an authentic manner using simple human factor techniques. Gangs ex- tends demonstrative identification (DI) to a group setting. The original allows users to indicate which two devices should communicate at a time, and is by nature designed for pairwise grouping. The basic idea of GAnGS is to use device pairing in an efficient manner for groups by using ancillary tools such as projectors for inputting information about the group (GAnGS-P) or by depending on other users in the group to perform tree- based pairing (GAnGS-T). Unlike our study, while GAnGS can be used to encounter-based attestation, it is primarily designed for collaborative data authentication. SPATE [23] improves on GAnGS by streamlining cryptographic operations to produce the system more functional on mobile devices. Neither work considers privacy or anonymity of participants, since authentication and collaboration are done at the same stage, and any possible attacker can keep participants in both designs easily by listening in on communication taking place between them. Connected to both deeds, although with slightly different applications, is SafeSlinger [17]. SafeSlinger emphasize usability when creating trust among participants in communication and on-the-fly collaboration settings.

Since placement is ace of the most often used bits of information for encounter verification, location proofs are studied in [32] and [21]. Some commercial platforms that ate- lies the idea of short-range communication and localization-based services include Brightkite [7] and Loopt [24] while other similar thoughts can be seen in WhozThat [5], Serendipity [16], SocialAware [18], Veneta [34], D-book [10], and Bump [8]

An application for contact data exchange that provides no privacy guarantees against a compromised central server; its security is analyzed and improved in [33]), among many others. 2 Most of these works do not consider location privacy, despite of its grandness.

Last system worth mentioning is MobiClique [30], which establishes an ad-hoc on-the-fly mobile social network by boot- strapping initial contacts from online (static) social network. As in our work, users in MobiClique use short-range Bluetooth communication and can establish encounter-based social links. Most interestingly, MobiClique provides several measurements demonstrating the feasibility of such system in terms of power consumption on typical mobile devices, as the one used in our conception. Yet, unlike our system, MobiClique does not guarantee nor address user privacy.

Overview of SMILE.

The primary work in the literature that is similar to our work in goals and purpose is SMILE. SMILE extends ideas from [26] to set up confidence between people who shared an encounter. It seeks to allow users equipped with mobile devices to establish such trust relation- ships while maintaining their privacy against potential attackers (e.g., the rendezvous server and other users in the encounter settings). In SMILE, users who want to communicate with each other must prove that an encounter took place between them. To answer this, the first device in the encounter generates and broadcasts the "encounter key" to other devices within its communication reach. The same device, then posts a cryptographically-secure hash of the encounter key, along with a message written in code using the encounter key to a centralized host. Referable to the pre-image resistance properties of the hash function, the centralized server cannot recover the encounter key without help, and thus cannot understand the message. Other users of SMILE with the same encounter key may claim the encounter by looking up the hashish of the key, which is used for indexing the encrypted message at the centralized host. Only users with the correct key will be able to decipher the message given by the first encounter party on the server, and every user with the right key can derive the retrieval hash value. The benefits of the basic design of SMILE as it is depicted here is that it brings down the misuse in the encounter system: only people who have been at the encounter place are those who recognize the encounter credentials and are able to claim the encounter.

In summation to the basic design, SMILE tries to provide two features: k-anonymity and decentralization. K-anonymity is achieved by truncating the hash values of the keys so that a single user is concealed amongst k other users with the same truncated value. SMILE features a decentralized organization that uses anonymizing network of re-mailers for communication, claiming to provide k-anonymity by requiring each user to possess at least k identifiers.

A plan that utilizes a centralized online entity. To boot, the claimed security guarantees might not conform to the prerequisites delineated above. While the confidentiality of encounter-related data is safeguarded by encryption, the secrecy of users in SMILE can be broken. Spell, in principle, the problem exists in organizations that rely on a centralized server, one can augment the performance of SMILE and mitigate the trouble by providing a server with high availability guarantees, which arrives at a cost that need to be taken as part of the plan. First, SMILE is prone to an impersonation attack performed by a user present during the meeting. Since no authentication is done during key agreement, any user can listen in on the encounter information and later claim to be the party of interest. This attack can further be extended to monitoring: if the adversary exchanges keys with the first user pretend- in to be the second, and repeats this with the other user, the adversary can take out a MitM attack and monitor all messages passed between users. Second, SMILE is prone to user collusion, an approach that was previously covered in social interactions [25]; a few malicious users colluding with the rendezvous server may possess enough data about the actions of other honest users (such as timestamps, locations information, and encounter keys) for the server to unmask users, finding out the identities of communicating parties. Lastly, while not particularly a specific problem of SMILE but every system using such a building block, the k-anonymity in SMILE requires that each user know the figure of other nearby SMILE users in order to make certain that there are enough people around to mask the activity of an individual— that the user is indistinguishable from k others in a given encounter setting. This, nevertheless, can be easily misrepresented by a Sybil attack [15] where a single adversary pretends to be ok − 1 other SMILE users, compromising honest user's
Anonymity.
E A comparison with SMILE. While we need a different approach than that used by SMILE, and any comparison between our approach and that of SMILE might turn unfair, we conclude this segment with the main differences in guarantees and functionality of our conception and that of SMILE. First of all, our design is scalable by nature, particular when considering the design option of using hidden services where users run their own servers for post-encounter communications. Second, our design provides stronger authentication features, by visual means, thus preventing the MitM attack, whereas SMILE does not provide these features. On the down side, the use of visual means for authentication is not universally accepted—see our user studies in section 6— and might have a privacy cost associated with it, while SMILE does not use such feature although at the risk of enabling the MitM attack. Finally, our designs, centralized or decentralized, provide better guarantees for the post-encounter phase by using a mix network to access encounter information, thus reducing the risk of giving away additional information to the potential adversary (impersonator or centralized server) by concealing networking information of users. While this feature can be added easily to SMILE, the fact that weaker authentication is used in it would still enable a variety of attacks the adversary can perform—e.g.,

It requires colluding with the centralized signing authority, while SMILE does not build use of such authority thus having the vantage of being lighter-weight than our pattern, only enabling the collusion in its attack surface.

IV. DESIGNS AND DESIGN OPTIONS

With requirements outlined earlier, we extrapolate the pattern of old arrangements. Particular care has been given to the protection and privacy requirements previous designs failed to accomplish. We split the design into functional blocks and identify potential attacks on several components of the organization. And so, we discuss two instantiations of the generic design; each with different benefits and trade-offs.

4.1 Functional Components

The functional purpose of a typical encounter-based social network consists of three major components located at three different architectural layers, as depicted in Fig. 1: user layer, plug-in layer, and "cloud." The term cloud may refer to a memory location of the encounters and private messages (e.g. A central rendezvous server or distributed "mini-servers") which is practiced by different encounter parties in the post-encounter stage. Nevertheless, the design can be quite flexible, allowing storage components to be dynamically chosen using a hack- in architecture: the system may support centralized servers, distributed hash tables [28], or even Tor hidden services [14]. Note that each of the different layers provides functionalities used to earn one or more functions or security requirement among these explained in §2. Furthermore, to make a balance between the functional and security demands, we also discuss two specific designs in the following subdivision. Infra, we elaborate on what requirements, each design meet.

4.1.1 On the Need for Strong Authentication

We have demonstrated in section 3 that simple authenticated key agreement during the encounter is vulnerable to a man-in-the- middle approach. Fed that the parties involved in the encounter are already cognizant of each other visually, the sole means to avoid this exposure is to apply a visual authentication system where users can acknowledge that they are communicating with the desired party simply by appearing at a photograph of that user. In other scenes such as a professional conference, a company logo and other information, which could be viewed as a reduced digital version of a line card (though, in many instances, the same scenario of using a personal photo on a personal business card still uses). To provide user authenti- cation, we take each user to receive a digital certificate signed by a trusted authority with sufficient information to identify users, including a picture of the user. The signing authority's public key would be known to all other nodes who use our encounter-based social net. It is not far-fetched to accept that future

authentication tokens such as passports and driver licenses will be released digitally, since cryptographic signatures make them more safe against malicious tampering than their physical counterparts. Though, we do not use such certificate but a modified one (see details in below). With that presumption, a user of an encounter-based system broadcasts a certificate

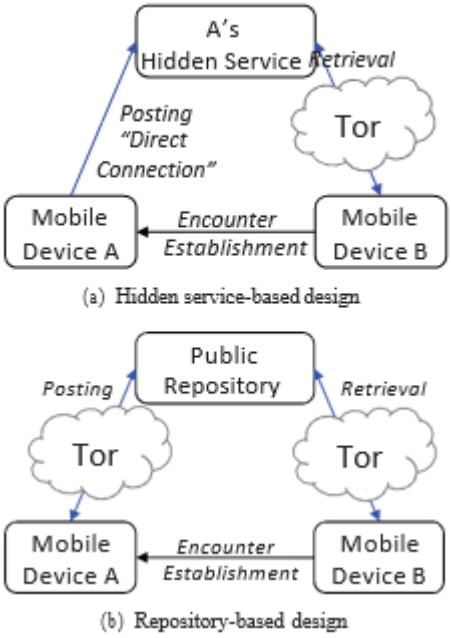


Fig. 1. Two specific designs. Fig. (a) illustrates the first design using Tor hidden services as encounter storage place. Fig. (b) illustrates the second design where users store encounter information on a public replica and gain anonymity to their access using a normal Tor operation.

with his or her picture and public key which is received by other people in the encounter space including the intended destination. Such information is then used for reconnection according to one of the design options explained in §4.2.

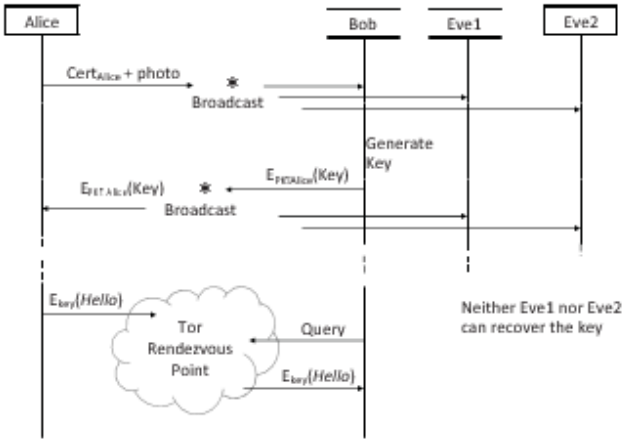
We note that facial recognition algorithms exist, which might reduce the privacy of the user, when an attacker collects photos from certificates being exchanged and compare them to photos associated with names and obtained from other sources such as other online social networks [2]. Although these attacks are computationally expensive, one may argue that the use of cheap cloud services may make these attacks very feasible. However, we answer this concern by pointing out three issues. First, even when using such cloud service, the attack, unless targeted towards a particular user, would be infeasible with a substantial cost that the attacker has to pay in order to breach the privacy of users who use our system. Second, the attacker does not need to collect broadcast certificates in order to apply the attack, but may simply take pictures of the encounter space and achieve a similar result to prove the presence of an individual at a certain place at a certain time. Finally, all prior work of facial recognition depends greatly on features extracted from original photos, but not from cartoon versions of them, which could be used to remedy the privacy breach associated with using a photo for visual authentication. 3 Our user study that considers cartoon version of photos instead of the original photos indeed hints on improved usability of our design.

4.1.2 Trusted Certification

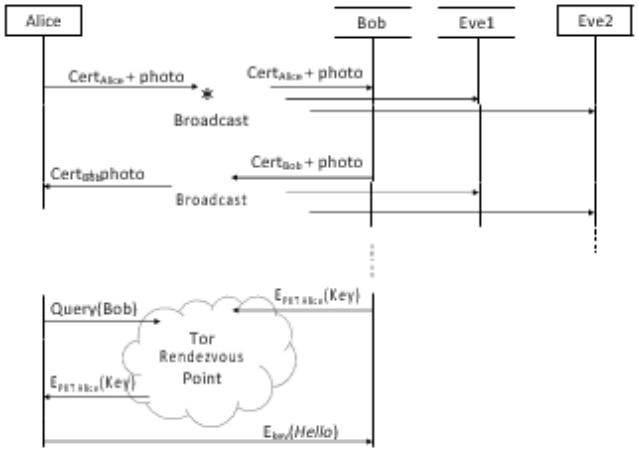
In our design, we use the X.509 standard [12] for certification without any modification to the structure of the certificate, but we limit the attributes available in the certificate used for encounters (discussed below) in order to preserve the privacy of our users. Indeed, the X.509 standard allows optional attributes for biometric information such as photos, which enables us to embed visual information into the certificate. The trusted authority mentioned previously is responsible for ensuring that the photo provided by user for certification is an actual representative picture, and allows others to visually identify the user. So, even when issuing a certificate that combines multiple pieces of private information, such as the certificate owner name and address, the authority will issue a separate, limited certificate with reduced amounts of private information which fits our social encounters design (only user's public key and photo). The ultimate signature by the trusted authority will sign all embedded attributes in the certificate, including the photo. Notice that the centralized authority used for signing the certificate with the photo does not need to be online for the protocol to work. Indeed, after the initialization phase of the protocol, in which certificates are issued for the participating parties in our design, verification of the signatures embedded in the certificate are verified at the side of the receiving party of the certificates using a publicly known public key of the authority. On the other hand, this guarantee comes at a cost that is not used in SMILE. Indeed, SMILE is lightweight since it does not require a certification entity, yet the certification entity provides stronger guarantees for the authenticity of participants. More details on the rationale of using such entity are provided below.

Our certification and visual authentication schemes are very simple. First, a user Alice generates a pair of public and secret keys (PK, SK), computes the hash value of her own image and other relevant information, including a Tor hidden service URI, which is a unique identifier that is used later by Bob to communicate with Alice over Tor hidden service. Alice embeds her PK and other metadata into a certificate request, and sends it to a signing authority. Second, the signing authority checks the validity of the metadata hashes in the certificate request and verifies the validity of the used attributes in relation with the previously mentioned extended certificate. If the verification process is successful, the signing authority signs the certificate using its own private key and sends it back to Alice. If at any time through the verification process any of the above conditions do not hold, the signing authority aborts and refuses to sign. Notice that here we omit some critical details: the authority only signs the certificate with the photo only if correctness of the photo associated with

the physical identity of Alice can be established, e.g. by physical presence of Alice at the authority.
At the protocol's run time, Alice broadcasts her certificate to everyone in the vicinity, along with the photo as credentials, which will later allow anyone present in the encounter space claim an encounter with Alice and proceed to the next phase depending on the protocol being used. In one of such design options — let Bob be one of the people to overhear the



(a) Immediate key exchange



(b) Delayed key exchange

Fig. 2. Sequence diagrams of our encounter-based social network design with two key exchange scenarios: (a) shows the immediate key exchange with postponed authentication and encounter reconnection via the Tor network, while (b) shows delayed key exchange and delayed rendezvous via Tor network (hidden service or direct Tor connections).

Broadcast, if the photo gives the visual authentication by Bob, Bob attempts to affirm if the certificate along with the photo are actual, i.e. hold a valid signature from a trusted agency. Bob calculates the hash of the photo and other information sent by Alice, comparing it to the value embedded in the certificate. Striking a match, Bob continues to verify the signature on the certificate by using the public key of the signing authority. If the signature is valid, then Bob admits Alice to be whoever she claims. Otherwise, Bob aborts. Note that this authentication procedure can be held over to post-encounter phase, as it is the case in delayed rendezvous.

4.2 Design Options

In the generic schemes outlined in §4. 1 we face two potential choices: do we command an immediate encounter key agreement between the two parties, or do we await? Each attack has a benefit and drawbacks. Immediate generation of an encounter key requires manual selection of the target user while still at the encounter level. Delayed generation, on the other hand, requires no immediate action along the theatrical role of the user, but potentially erodes user privacy during later communication. Both of these methods are discussed further under. Notice that these are not options to be taken within a individual organization; this alternative must be made before deployment to have a consistent protocol among all users in the network.

4.2.1 Immediate Pairing

If a user is willing to manually choose the picture of other users of interest while still at the encounter site, she can compose an encounter key, encrypt it to the selected user's public key, and send the resulting message. Each user in the vicinity will detect the transmission and try to decrypt it. Nevertheless, only the target user will be able to decipher the message correctly, and so recover the encounter key. This key will be applied afterwards to exchange secret messages at the rendezvous spot. This method prevents the rendezvous server

And colluding adversaries from determining which two users are transmitting. We can move a step farther and use timed-release encryption [31] to cover the contents of the message even from its intended recipient until the meeting is complete, ensuring that users do not inadvertently give themselves away by using their devices at the same time. In principle, time-release encryption would allow A sequence diagram showing the procedure of this key generation design is in Fig. 2(a).
While the advantage of this design option is enabling users to make decisions while at the encounter space while they remember well parties they encountered, enabling direct communication, and use of the physical encounter, reasoning about some security guarantees in this scenario might not be equally comfortable. Particularly, unconventional attacker capable of measuring signal strength, and associating that to users might be able to break the secrecy of users by matching who meets whom by monitoring the encrypted traffic between them, therefore violating the unlinkability requirement.

4.2.2 Delayed Rendezvous

Devices will consistently broadcast their certificates, but will not require others users to instantly survey the data. (Every bit in the immediate pairing scheme, we can use timed-release encryption [31] to enforce this constraint.) At a later time, the device user can look at the list of collecting identities (and public keys) and select those with whom he likes to

convey. As before, we will utilize non-malleable encryption to compose a message to the other user, but instantly the message must be salted away "in the cloud" in such a manner that it is linkable to the public key of the user for whom it is intended, and some encounter nonce passed at the time of the meeting. This may not be a significant problem, thinking that only keys and faces are uncovered, and not more personal components of users' identities. A sequence diagram showing the functioning of the two key generation design options is presented in Fig. 2(b). While this system does not suffer from the shortcomings in the immediate pairing scheme, the capability Of reconnecting to encounter parties depends entirely on the capability of encounter parties to call in such encounters. We believe remembering people are quite easy, given the special number of encounters per time window.

4.2.3 Decentralization and Anonymity

Our distributed design, one that does not need a rendezvous server, is portrayed in Fig. 1(a). We apply the generic design described in §4. 1 combined with Tor hidden services [14] to provide communication anonymity. While Tor provides users with anonymity, Tor hidden services enable servers to hide their identities as good. Each user is given his own Tor hidden service and utilizes it for two purposes: first, to conceal his identity and gain anonymity as to his location and second, to serve follow-up requests relating to previous meetings. The other party must use the Tor client to access the covered service, also gaining anonymity and hiding her location from the waiter. This pattern can easily scale to a large number of concurrent users [22], and is resilient to failure, since an approach on the entire social network built using this distributed design would call for attacking many individual nodes simultaneously (i.e. The bankruptcy of one hidden service would not affect other hidden services).

4.2.4 Centralized Design with Anonymity Guarantees Our second design is depicted in Fig. 1(b). Here we go into a public repository to which users involved in the

Can post encounter information. Say that Alice shares a public space with Bob, and therefore takes his public key from his certificate. At an arbitrary time after Alice and Bob share a location, Alice can go through all her collected identities, notice Bob's picture, and determine to impress up a conversation. She composes a message to Bob, encrypts it under Bob's public key, and posts the encrypted message in the centralized repository under Bob's public key. To gain anonymity as to her identity and position, Alice uses a Tor client, concealing her IP address from the central host. This is more efficient than the covered services used in the previous protocol, which require one of the encounter parties to be online all the time to serve other parties involved in the confrontation. In this purpose, on the other hand, Bob can get the messages left for him at the central repository at any time. He similarly accesses the repository through Tor to hide his identity, and downloads all messages directed to him. To identify such messages, we suggest using nonces as part of the indexing system. These random one-time values, generated and exchanged at runtime of the encounter protocol, along with the public key of the encounter party that initiates the meeting, are hashed and used for indexing. By doing so, a malicious repository will not be capable to find any data about the identity of the individual accessing the repository unless at least one individual at the encounter site is malicious and colluding with the secretary. Note that the use of Tor here is not to uphold the privacy of the participants from an adversary present at the meeting, merely from the storage server himself. The use of Tor conceals the participants IP address, and thus location, and disable breaching privacy by associating participants with such data. It is observed, nevertheless, that such collusion with the server is not possible when using Tor's hidden services, since each participant works as his own server; although this choice arrives at some cost by requiring participants to be online

V. IMPLEMENTATION AND EXPERIMENTS

To validate our method and assess the practicality of our purpose, we enforced the system on the iPhone platform and proved it on multiple devices under ideal conditions, as well as conditions that users are likely to encounter in urban settings- tings. In our implementation, we used the delayed rendezvous scheme where the user's device can collect simulated broadcast information during encounters and then use the decentralized Tor hidden service architecture for the second component of the meeting. Those take a hidden service URI (an address through which one can access services deployed by hidden servers [14]) to be part of the user's information and is therefore associated with the certificate as a bundle in sent the transmissions. Note that, even when an adversary captures the certificate exchanged between two honest participants, and obtain admittance to the URI, the honest participant running the hidden service will still experience a full control over whether to react to requests for communication sent via the hidden service. Consequently, while the role of the hidden service would resolve the rendezvous problem and provide the means for reconnection in the future based on the previous encounter, it will not increase the attack surface by enabling means for the adversary to break the

Privacy of the users and their encounter.

Lastly, notice that our pattern is generic. We are not restricted to, any specific platform like Apple's iOS, which we chose for development, in any of the our design ingredients. Our choice of development platform for our proof-of-concept application is simply due to availability and ease of use for quick prototyping. Other programs, such as Android, would work precisely as good. Therefore, any conclusions on the serviceability of our design are independent of the platform, as we merely demand a smart phone with basic wireless capabilities.

5.1 MeetUp: An iPhone Application

Our iPhone application, called "MeetUp," allows users to find other users of the system within Bluetooth range, settle with whom they like to communicate, and beam and receive secret

messages. Screenshots of typical use scenarios are presented in Figures 3 (a) through 3 (c). The user looks for other nearby users of our system, and receives their identification information, including photographs and certificates signed by our trusted certificate authority.

5.1.1 Certification and Visual Authentication

The certificate authority uses a scaled-down edition of the architecture shown in part 4. Certificates signed by the authority include hashes of photos and Tor hidden service URI unique to the user. The file containing the certificate, the picture, the hidden service URI, and the signature are the deployed to each device in the arrangement. The certificate authority is responsible for verifying that only single case of such file is deployed per user. It is likewise responsible for verifying that the biometric authentication

used in modern passports [13].A larger deployment of our system could rely on an already- implemented certificate infrastructures that use photographs, such as a driver's license records, as discussed in §4.1.

5.1.2 Wireless Communication

Inter-device communication was implemented using Blue-tooth [6]. The special reach of Bluetooth devices ensures that users are within close physical proximity to exchange certificates. This earns it more likely that users are within visual range and can identify each other. For the delayed key exchange, we rely on the fact that humans can easily pick out a font that has been seen before [19] when we represent multiple devices that have been discovered previously, along with photos relevant to the proprietors.

Our next step was to enforce the broadcast protocol over Bluetooth. Unluckily, the Bluetooth specification does not explicitly support broadcast in the way we want. One broadcast scheme allowed by the Bluetooth standard is over a piquant [6] in which a minor number of devices within radio range can make a temporary ad-hoc network. Broadcast communications then take place over those small networks. Unluckily, the Apple SDK does not support piconets in iOS 4.1 [4] at the time of composition of this report. The only choice on the iPhone platform was to use Bluetooth peer communication, using repeated unicast to emulate broadcast. It holds a major drawback since both parties in a peer session will accept to acknowledge each other's device before any data exchanges can take home. This will obviously cause problems for the delayed message setting, but in our example it was sufficient to obtain some RF measurement data for our experiments. As luck would have it, since we strictly emulate broadcast, this forced implementation choice does not breach our security guarantees. Bluetooth sniffers and strength indicators might be capable to assist in localizing a transmitting device, but that data is already presumed to be public. Equally for the recipient, if a user relates to every other device in the vicinity one by one in random order and exchanges equal amounts of data with all of them, an adversary cannot determine the designated receiver.

Observe that the Android platform enables broadcast [1], and a potential final product of our design could be earned on the Android platform—although, as noted before, the main limiting factor for using the iOS platform at the time of composing this work was the availability of IOS devices to develop the application and to test the invention. Remember that changing the platform will have no impact on the cogency of the respite of the effects, especially the user fields, since recent independent surveys have demonstrated that the Android OS is taking dominance as it is used on 52% of the smart phones in the US, as opposed to 35% for the iOS [11]. A deployment on an Android device would take a single message for a single broadcast, instead of the simulated scenario described above which is necessitated for demonstrating the basic idea with existing equipments.

(a) Searching

(b) Users found

(c) User details

Fig. 3. iPhone App implementation screenshots

5.2 Using MeetUp

The first step in utilizing our application is for the user to begin scanning for devices within Bluetooth range. The

applications on remote phones have to be taking heed as well start a Bluetooth peer session. In one case the session is set up, the two devices can exchange certificates, photos, and signatures. With this method, nevertheless, the initiating device has to work through the list of nearby devices and practice an individual pairing with each remote device in turn instead of performing a real broadcast, adding considerable overhead.

In the following measurements, we weighed the time required to transfer a 20KB bundle between two paired de- vices. We have brushed off the pairing time since it was a step necessary only for our emulated network. In a real broadcast, there will be no pairing time. We did not receive access to the proper equipment (such as a spectrum analyzer) to actually quantify the amount of traffic on the 2.4GHz band, and then we picked out locations with minimal RF interference and densely populated regions around the campus with a heavily utilized 2.4GHz band for urban contexts.

5.2.1 Effective Range

We applied an open field in a sparsely populated area to obtain ideal condition measurements. Such an environment ensures minimal interference over the 2.4GHz Bluetooth communica- town band and minimal multi-route due to signal reflecting off of targets around the communicating devices. Our experiments indicate that under those conditions, the devices can reveal each other and exchange data at a range up to 24 ms. Transfer times increased as we increased the space between the two devices, but all were faster than 400ms (shown in Figure 4). We also counted at the direction of the communication to determine if users have to be showing their devices in a peculiar direction to ensure timely transfer of data. We quantified the time taken to transfer 20KB of data over our Bluetooth channel from a user obtaining a device in a special fashion. Measurements were taken at 45◦ increment by a querying device moving around a responding device. The experiment was repeated for radii of 1, 2, 3 and 4 meters or so the responding device. We did not find any significant transfer time differences in all of our measurements. The average transfer time was approximately 250ms for the 20KB payload (the results are depicted in Figure 5).

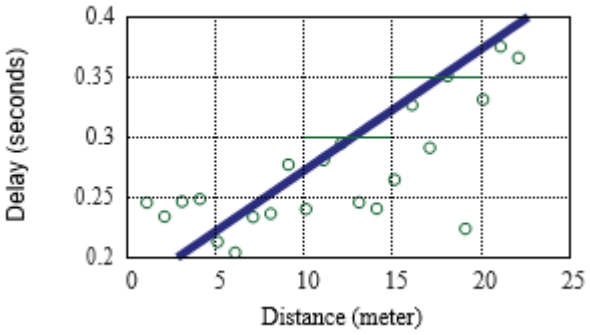


Fig. 4. Delay as the time it takes to send encounter information (about 20KB) and receive it by other encounter parties with variable distance without obstacles.

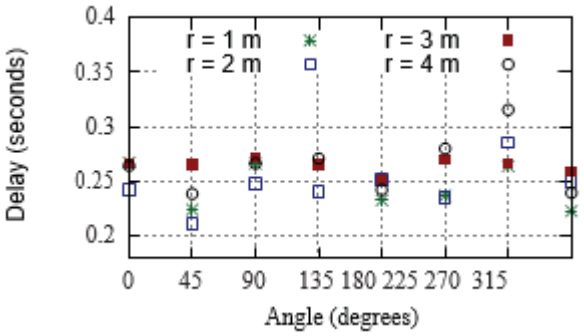


Fig. 5. Delay with different locations of the encounter sender and receiver, as determined by a radius r (in me- ters) and an angle θ (0 to 325 with 45 degrees increment).

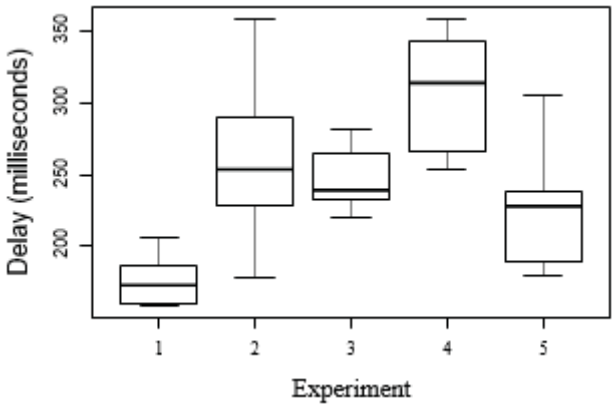


Fig. 6. Delay with several scenarios representing different potential deployment settings of MeetUp.

5.2.2 Effective Range with Obstacles

We look at the time assumed to transfer and receive encounter information between two encounter devices under several conditions reflecting real-world deployment settings, where obstacles around the encounter parties may cause signal attenuation and multi-path noise. We consider five communication scenarios of interest: (i) through a barrier (door), (ii) in the hallway—line of sight with a detachment of 20 meters, (iii) communication across multiple walls, (iv) while on different floors (2 floors separation), and (v) when one of the parties is in an elevator and the other is outside it. For each scenario, ten measurements are studied and the consequences are shown in Figure 6, where we plot the 5 representative values of min, max, median, Q1 and Q3 (lower and upper quartiles). While some scenarios imposed far greater delay than others, the data generally indicate the feasibility of Meet Up in several potential deployment settings.

5.2.3 Measurements in Urban Settings

We tested MeetUp in a thickly-populated urban setting, in a bus station populated by students equipped with mobile phones, with this being as the only difference from the range

and obstacles experiments above. The information gathered from this experiment are presented in Figure 7. We notice that it requires less than a second in all cases to do the encounter, and on average it

0
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

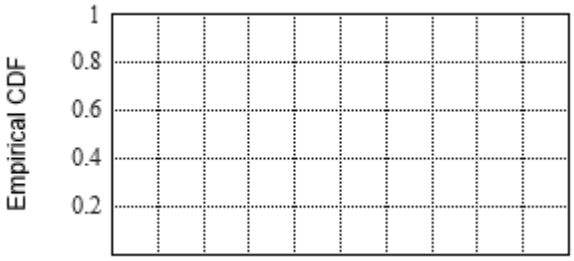


Fig. 7. Delay as the time it takes to send encounter information (certificate and data in a 20KB bundle) and receive it by another encounter parties in urban contexts

Holds about 600 ms While larger than an environment free of obstacles, multi-path and interference, the transports were still finished in an acceptable time window, therefore holding up the practicality of our plan. For this experimentation, only two users participated in the experiments where one is the initiator of the encounter and the other is the receiver. The results presented in Figure 7 correspond to roughly 100 versions.

5.2.4 Tor Hidden Service

Sticking with the device encounter and data transfer over the wireless net, we used a Tor hidden service for the second form of the anonymous encounter. We transferred a 40KB data bundle that only the designated receiver will be able to decipher. We made a new Tor circuit for each experiment, and we ran multiple timing measurements per experiment. The timings tend to be very consistent per circuit, but very different between circuits (ranging from 1.5 to about 8.5 seconds). Most circuits we used to demonstrate an acceptable transfer delay of under 10 minutes. Observe that the Tor hidden service does not increase the attack surface, but rather hides the users' additional network information, such as IP address, while enabling the rendezvous in a decentralized manner.

5.2.5 Technical Issues

During our tests we discovered that a number of users tend to utilize their actual names as their devices' names due to the default naming strategy used by the mobile device. The device name (which in many cases carries the user's name) is communicated to other devices during the Apple iOS Bluetooth pairing protocol. Such a naming scheme would defeat the intent of the anonymity offered by the protocols outlined in this report. In future implementations we intend to utilize the fourth dimension of the meeting as a selector, or else of a device's nickname. From a user's degree of view, the appropriate photos will be exhibited next to the time of encounter allowing proper selection. The tilt can also be augmented with locations at which the meeting took place.

In our execution, we exported the certificate from the twist to a desktop which then made a rendezvous point over Tor. There is also an option to deliver the device itself connect to the Tor network to place up a rendezvous spot. At the time of composing this paper, the iPhone Tor client required a jailbroken iPhone, limiting its usefulness. Depending on the policies of the iTunes store, we may be able to include a Tor client component within our application, or rely on a connection to a desktop computer to make the rendezvous spot and wait for incoming links. Other future implementations of MeetUp would consider other potential mobile devices that already support Tor, e.g., Android platforms [3]. Note that if one is to look at the iPhone platform to serve as the platform of MeetUp, one simply needs to yield up the option of running hidden services over the iPhone while other options are, including running hidden services on a desktop, would be still available and are independent of the platform.

VI. EVALUATION AND DISCUSSION

6.1 Privacy Evaluation

6.1.1 Privacy in the Encounter Phase

In the first phase of the encounter (when users are still in the same location), the first party referred to as the encounter source uses a broadcast communication channel that gets the second party of the opposition which we refer to as the encounter destination unlinkable to the source. Information broadcast by the source is received by every other party in the encounter space, and no destination information is disclosed. The only information disclosed about the source is her public key and photograph. We discuss the privacy implications of this setup below. Nevertheless, it is clear that while an adversary present at the meeting can determine who else is present and using MeetUp, the adversary cannot determine if any two users made a link. In the post-encounter phase and for the role of reconnecting with users who were present during an encounter, the identity and location of the individual initiating the connection to the rendezvous server or hidden service are obscured using the Tor network, or Tor hidden service, though in the latter example is immediately exposed to the source of the encounter if verified.

6.1.2 Privacy in the Post-Encounter Phase

While it is easy to reason about the second case where a user is fed his own hidden service, since the surety of the communication is inherited from that of the Tor network and computing entities under the full control of the user, it is more difficult to determine whether unlinkability holds when using a centralized rendezvous server. Since encounter information is deposited on the central server of the destination and is grounded on the source's information (e.g. An index derived from his public key), this data might be used to violate the secrecy of users any entity may contain in the source's mailbox to see if there is a message. Remark that the message is encrypted, and therefore still confidential. The problem is

alleviated using encounter-time random nonces, which would be merged with the destination party identity to derive the rendezvous key used by the encounter parties.

But a malicious server colluding with a user in the en- counter space would be able to pull any data from the rendezvous key the server acting alone makes no useful data. Furthermore, the extracted data is limited to the time and space of the skirmish, and naught else. The certificate of the beginning, if the destination decides to report it as an evidence of the brush, is encrypted under the source's public key along with the message left in the root at the fundamental host. Such information about the root cannot be tied to any other information, since the server in our designs does not store such information, unlike the case of SMILE. Note that another reason for using per-encounter nonces generated at encounter time is to optimize the communication overhead when retrieving the encounter information. A fix for the colluding server and malicious users mentioned above is to create each user who desires to retrieve encounter information posted to him on the server to perform "dummy" queries to disguise his intended encounter query.

6.1.3 Privacy Concerns due Visual Authentication

I may criticize our design for practicing a personal photo associated with the encounter information, which may be eavesdropped by all users in the meeting scene, including the attacker. While the photo-to-key binding may be abused to degrade the privacy of users, we contend that this is a necessary piece of information, and a potential attacker might take it from several other sources, apart from this application. We further argue that such data is already available to the attacker by physically co-locating with the encounter party, and by ascertaining who is present at the same position in the same position. Nevertheless, we emphasize that this data cannot be used to break the privacy guarantees of the encounter, since the adversary cannot read messages exchanged between users, nor does he recognize the identity of the other party in the skirmish. We finally argue that users interested in maintaining unlinkability provided in our intent for their clashes are also willing to pass this bit of private information away, for the ultimate benefits gained. This claim is further confirmed by several user studies through surveys.

We emphasize once more that our scheme requires a visual authentication to avoid certain security breaches, namely the MitM attack, while SMILE does not (at some security cost).

Our end is to insure that we are indeed sending messages to the appropriate party. At the tip of a short encounter, the users do not have any info about the recipient, except for a visual information, which is driving our require- meant for visual authentication. This requirement does have a privacy angle to it, but in the historic period of public Facebook profile pictures, users are finding it more satisfactory to possess pics of themselves available, which is—as pointed out earlier— already available through other sources. This includes, as an analogous case to the technique used in our work, having a video camera running during the train ride and catch a picture of everyone. More details on the usability associated with this building component are highlighted in the chase.

E Survey settings. To show the usability of our intent,

We perform several user studies as presented in the subsequent paragraphs. In these user studies, we recruited 76 volunteers with age ranging from 21 to 35 through a Facebook application designed specially for this work. Invitations to take part in the survey are made to the sample size from a larger pool of candidates, of approximately 320 subjects, and each candidate in the sample is chosen from the pool uniformly at random. The mean age of the respondents to the survey was 25.7 years old, with 45 of the respondents being males while the rest being females. Of the responding subjects, 68 had at least a bachelor's degree (or were enrolled in a course of study that leads to a bachelor's degree at the time of the survey; about half of them are enrolled in or had a graduate level) while the rest suffered a two years or less of education past high school. Today we continue to describe these user studies in more details.

E User Study 1: Using Photos for Authentication. To Understand the potential of our intent in real contexts, we perform a case study on a random sample of 76 subjects described above and examined their willingness to apply their personal photographs as part of an authentication method in order to improve their privacy. Away of the 76 subjects, 4 subjects did not respond (correspond to 5.26% of the sample size). Thirty six (36) subjects responded positively by agreeing to use their photos (correspond to 50% of respondents and 47.37% of the overall sample size) while 22 responded negatively (correspond to 30.6% of the respondents and 28.95% of the sample size) and 14 (18.42% of the respondents and 19.4% of the sample size) selected to turn the feature on at times. In total, 50 out of 74 respondents (correspond to 69.4%) are likely to use the feature by providing their personal photos for certification and authentication when using the service.

We repeat this user study, but at this time by asking the same subjects whether they would be willing to use our application if their photos were to be replaced by a cartoon version as a remedy to privacy concerns. Out of the subjects who negatively responded in the earlier study, 8 indicated it is likely to turn on the application at times, while 4 indicated their willingness to use the application. 10 (correspond to 13% of the population) responded negatively. In total, 87% are likely to use the feature by providing their personal photo or a cartoon version of it.

6.2 Overhead and scalability

The overhead required in MeetUp is in the form of com-

Sources are required for transferring and receiving encounter information, computations are involved for establishing Tor circuits, in normal and hidden-service based operation, and memory is required for storing the encounter information in the mobile device and subsequently on a desktop machine that is employed for carrying the hidden service. While both are conceived for the resource requirements, of interest to our feasibility study is the mobile device used for carrying out the

encounter operations. Here, we verify the feasibility of MeetUp and its reasonable consumption of resources.

Equally we have indicated in the previous section, the time it requires to exchange, encounter information in our blueprint is small, and in most cases is less than 1 second on typical devices. Furthermore, in many of the deployment environments that we have seen, this overhead is even around 250 milliseconds, making it really viable to employ.

The memory required in our figure, per encounter, and shown earlier in §5, is about 20KB. While this is great in relation to previous memory consumption requirements for similar purposes, such as SMILE, we think that this is sane for the provided guarantees, and caved in the amount of resources in many of the current smart phones which are equipped with GBs of memory. For instance, with a 512MB allocated for the application, one may store up to more than 25,000 meetings. Passed that one holds the option to decide to store the encounter or discard it right away, this distance of memory can be further applied to lay in more useful meetings. As well, fed that the offline communication and reconnection is performed through a non-mobile machines, as advised by our purpose, this memory can be further used in a more honorable way: the memory required on the mobile phone is only for fresh encounters, which are limited per days [27]. On a desktop machine, 1GB of memory is plenty for storing 50,000 encounters per user, far more than the number of friends one can realistically deliver.

The computations in our plan are generally cheap to perform on typical mobile devices. The only online computations re-cured in our design is a signature verification in order to affirm the authenticity of certificates issued by the certificate authority. This overhead can be further minimized by considering verification for encounters that go along the visual authentication, or can be further moved to non-platform in an offline phase. This conclusion, nonetheless, may or may not be desirable based on the tradeoff set by users between computations and memory consumption (as one needs to lay in all encounters, including undesirable ones, in parliamentary procedure to perform verification offline). In sum, the computation involved in our design is reasonable and feasible for most mobile devices.

On the other hand, offline computations and communication required for our application are different from those required on the mobile telephone. While memory requirements are still same, in the offline phase, we use Tor, or Tor hidden services, to provide concealment. By measuring that for the same measure of communication overhead (20KB), we found the time it requires to transport such information over Tor (using previously established circuit) is roughly 3 minutes. This further confirms the feasibility claims of our plan. I may contend against the usability of MeetUp, given

Typical smart phones limited batteries which may run out quickly due to the hard usage of Bluetooth communication. Still, we notice that even when one keeps MeetUp running all time and scan for encounters every two minutes, typical smart phone battery would serve for more than eight hours, as it is demonstrated in [30] with MobiClique, in which the operating cost is comparable to the overhead in MeetUp.

6.3 Usability Issues

Our design assumes the availability of smart phones for users and their willingness to utilize their phones to take part in the scheme. To infer the density of smart phones and willingness of people to practice them in our diligence, we perform the following user study.

E User Study 2: On Using Smart Phones.We examine the record survey outcome on whether subjects are willing to use their smart phones for applications such as MeetUp or not. In the same sample, 25% of the questioned subjects did not respond, implying the likelihood of not having smart phones or not willing to utilize their telephone sets for social networks for such appli- cations as MeetUp. Nevertheless, 39.47% of the sample (52.6% of the respondents) answered positively, 28.95% (38.6% of the respondents) answered negatively, and 6.58% (8.8% of respondents) answered with “maybe” for the likelihood of using their smart phones to get in touch with people they see. Out of 57 respondents, 35 (correspond to 61.4%) are likely to use smart phones for connecting to people they meet in MeetUp.

E User Study 3: On the Density of Wireless Gadgets. The typical usage case for the MeetUp application is in a social context where people congregate. To this goal, we selected a library on the campus of a major North American university. Since MeetUp runs on the Apple iOS platform, we plan an experiment to gauge the density of devices capable of extending our software in the chosen position. Note that this user study does not prove whether these devices’ owners are willing to use our application or not, since this is already proven in the two previous user studies. Consequently, the bulk of the devices did not deliver our application installed. Presented the results of our two previous case studies, one can understand that the bearing of such population of gadgets at any time in the suggested deployment scenario would create a sound conclusion on the serviceability of our diligence.

Apple makes a service inside their devices that help in zero configuration situations. For this service, the devices use the name specified by the user for communication between devices on a local net. By default, that figure bears the type of device it is. The first step in this protocol is a Multicast DNS query on the local network to check for name collisions. The devices run this protocol by default upon first joining any Wi-Fi network.

We plugged in a laptop to the local Wi-Fi network at our selected position and listened for Multicast DNS messages. To insure that we were confined to the target position, we threw a user with a known device name connect to the same network, but at a different position, and verified our inability to observe his device’s DNS messages. We calculated the area of the target location to be around 5000m2.

We collected messages heard on the Wi-Fi network for 5hours, and filtered the Multicast DNS queries. We then pressed out the unique IEEE MAC addresses of the querying device from

those messages, and got rid of any duplicates. We verified that all the MAC addresses belong to the IEEE OUI prefixes assigned to Apple to filter out any devices with an iOS name, without being one. Those default self-assigned names identifies the device type as "iPhone", "iPod" and "iPad", leaving us to estimate the iOS device diversity.

To enable a social encounter with mobile devices, it is significant for those devices being at the same location at the same time. Observing Multicast DNS messages tells us when an IOS device joins the network, but we don't know how long it sticks around. We can get a rough approximation based on the physical attributes of the emplacement. With the radius of our location being 35m and the average human walking speed of 1.4m/s, we approximate that a user will stay in the network for at least 25 minutes. Using the Multicast DNS messages, we can calculate a lower limit for the number of devices coming online at the above time intervals. In our results shown in figure 8, we counted the number of devices announcing on the network within 25 second buckets. We filtered out duplicate DNS requests, and replies originating from the bearer within the reply timeout window to avoid double counting devices. On average, we observed about 9 devices joining the net every 25 minutes. The measurement started around 1pm local time, which would explain the initial bump in devices, traced by a gradual descent in what would be dinner time locally.

The limitations of our counting technique include the tracing. The the iPhones have to be configured to connect to the university's Wi-Fi network. While not counting all devices, this is a plausible case for the Wi-Fi data network is a lot quicker than the 3G network on campus, therefore users have an incentive to turn Wi-Fi on. (ii) The names of the devices can be changed to get rid of the device type. We don't know the fraction of user who would do so, but we induce at least a lower bound on the number of Apple IOS devices.

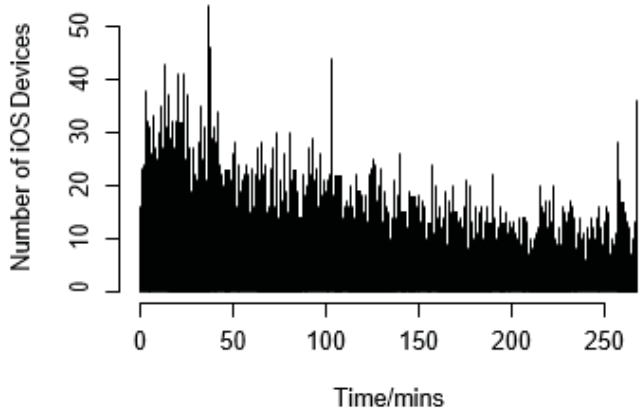


Fig. 8. Apple iOS device density estimation

Even with those limitations, we were able to observe 448 unique devices, including 257 iPhones, 129 iPods and 62

IPads on the net within 5 hours in an expanse of 5000 m2. Measuring the Multicast DNS messages indicated their presence along the mesh, possibly at different points in time as the device sleeps and wakes up while staying in the same geographic position. This density of devices provides us with some confidence that the MeetUpapplication could be useful in augmenting a social network graph based on geographically proximate social encounters.

6.4 Additional Applications

Today we turn our attention to further applications. We elaborate on applications mentioned in §1 that may call for anonymity for shared encounters. Among many others, we discuss two instances.

6.4.1 Key Distribution

Key distribution is a challenging problem in the context of distributed computing schemes. One obstacle for the key distribu - town is the fact that it is difficult to create an authority always online to take care of the distribution of keys, as easily as the scalability issue of central distribution for larger meshes. Our proposed design can be utilized for key distribution, and can be applied as a plug-and-play service for this intention. For instance, look at the application of OneSwarm in [20]. In OneSwarm, there are two strata of users, trusted and untrusted users, and both are practiced for dissimilar purposes and differ in the way they receive the keys and their function in OneSwarm. While the trusted users get their keys from those who trust them right away in an "offline" fashion, untrusted users get their keys from a key distribution center, that should be online all the time. Using our design, one may distribute keys to interested users based on activity shared with them such as an encounter. I even may consider the scenario of establishing trust based on the encounters [23]. Other key distribution applications that may benefit from our design include storage services, file-sharing, and so on.

6.4.2 On-the-fly Name Card Distribution

Take the scenario of scientific meeting, where some researchers present their work, some others take part in discussions on the work, and none has time to keep in touch and introduce him to all researchers, due to the time restraints. Our application can be brought into action for such scenario for on-the-fly name or business card distribution. Again, same as the main motivation of our application, people are adept at remembering faces of another encounter people rather than names, and hence it is easy to connect a digital name card associated with a photo than that of remembering names.

VII. CONCLUSION

In this study we demonstrate that existing plans for secure encounter-based social networks fail to fulfill reasonable sea-charity guarantees. We outline several requirements that ideal encounter-based social networks need to fulfill, and introduce a generic framework for constructing encounter-based social nets. We then apply our theoretical account to showcase several Patterns, and march that our designs fulfill more

requirements than SMILE, the design the motivates our study. We demonstrate the feasibility of our work through a demonstration of MeetUp, an iPhone application that uses our design. In the future, we will investigate further extensions to the current framework, alternative design options, and additional pluggable components. We will also investigate developing MeetUp on other mobile platforms as well as a large-scale deployment using multiple wireless communication protocols.

REFERENCES

[1] "Android Broadcast Documentation," http://goo.gl/FTxzV.

[2] A. Acquisti, R. Gross, and F. Stutzman, "Faces of facebook: Privacy in the age of augmented reality," in BlackHat, 2011.

[3] "Android development kit," http://developer.android.com, October 2010.

[4] Apple Inc, "Apple iOS Networking & Internet," http://developer.apple.com/technologies/ios/networking .html, October 2010.

[5] A. Beach, M. Gartrell, S. Akkala, J. Elston, J. Kelley, K. Nishimoto, B. Ray, S. Razgulin, K. Sundaresan, B. Surendar, M. Terada, and
R. Han, "Whozthat? evolving an ecosystem for context-aware mobile social networks," IEEE Network, vol. 22, no. 4, pp. 50–55, 2008.

[6] Bluetooth, "Bluetooth Specification Version 4.0," Bluetooth SIG, 2010.

[7] Brightkite, "http://brightkite.com/," October 2010.

[8] Bump, "iPhone and Android application," bu.mp/, 10 2010.

[9] C.-H. O. Chen, C.-W. Chen, C. Kuo, Y.-H. Lai, J. M. McCune, A. Studer, A. Perrig, B.-Y. Yang, and T.-C. Wu, "GAnGS: gather, authenticate 'n group securely," in MOBICOM, 2008, pp. 92–103.

[10] R. J. Clark, E. Zasoski, J. Olson, M. H. Ammar, and E. W. Zegura, "D-book: a mobile social networking application for delay tolerant networks," in Challenged Networks, 2008, pp. 113–116.

[11] CMS Wire, "Android dominates burgeoning us smartphone market," http://goo.gl/WZ4tZ, August 2012.

[12] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (IETF RFC 5280)," Internet Engineering Task Force, Request For Comments, 2008.

[13] "COUNCIL REGULATION (EC) No 2252/2004 of 13 December 2004 on standards for security features and biometrics in passports and travel documents issued by Member States," Official Journal of the European Union, December 2004.

[14] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second- generation onion router," in Proceedings of the USENIX Security Sym- posium, 2004.

[15] J. Douceur, "The sybil attack," P2P Systems, pp. 251–260, 2002.

[16] N. Eagle and A. Pentland, "Social serendipity: Mobilizing social soft- ware," IEEE Pervasive Computing, vol. 4, no. 2, pp. 28–34, 2005.

[17] M. Farb, M. Burman, G. Chandok, J. McCune, and A. Perrig, "Safes- linger: An easy-to-use and secure approach for human trust establish- ment," Carnegie Mellon University, Tech. Rep. CMU-CyLab-11-021, 2011.

[18] C. M. Gartrell, W. C. M. Gartrell, D. S. Mishra, S. Charles M. (m., and C. Science, "Socialaware: Context-aware multimedia presentation via mobile social networks," 2008.

[19] P. Hancock, A. Burton, and V. Bruce, "Face processing: Human percep- tion and principal components analysis," Memory and Cognition, vol. 24, pp. 26–40, 1996.

[20] T. Isdal, M. Piatek, A. Krishnamurthy, and T. E. Anderson, "Privacy- preserving P2P data sharing with OneSwarm," in SIGCOMM, 2010, pp. 111–122.

[21] V. Lenders, E. Koukoumidis, P. Zhang, and M. Martonosi, "Location- based trust for mobile user-generated content: applications, challenges and implementations," in HotMobile '08: Proceedings of the 9th work- shop on Mobile computing systems and applications. New York, NY, USA: ACM, 2008, pp. 60–64.

[22] J. Lenhard, K. Loesing, and G. Wirtz, "Performance measurements of Tor hidden services in low-bandwidth access networks," in ACNS, ser. Lecture Notes in Computer Science, M. Abdalla, D. Pointcheval, P.-A. Fouque, and D. Vergnaud, Eds., vol. 5536, 2009, pp. 324–341.

[23] 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, pp. 1–14.

[24] Loopt, "http://loopt.com/," October 2010.

[25] M. Macy, "Learning to cooperate: Stochastic and tacit collusion in social exchange," The American Journal of Sociology, vol. 97, no. 3, pp. 808– 843, 1991.

[26] 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 '09: Proceedings of the 10th workshop on Mobile Computing Systems and Applications. New York, NY, USA: ACM, 2009, pp. 1–6.

[27] J. Manweiler, R. Scudellari, and L. P. Cox, "SMILE: encounter-based trust for mobile social services," in ACM Conference on Computer and Communications Security, E. Al-Shaer, S. Jha, and A. D. Keromytis, Eds. ACM, 2009, pp. 246–255.

[28] P. Maymounkov and D. Mazie`res, "A peer-to-peer information system based on the XOR metric," in IPTPS, . I. P. of the 1st International Workshop on Peer-to Peer Systems (IPTPS02), Ed., 2002.

[29] A. Mohaisen, E. Y. Vasserman, M. Schuchard, D. F. Kune, and Y. Kim, "Secure encounter-based social networks: requirements, challenges, and designs," in ACM Conference on Computer and Communications Se- curity, E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, Eds. ACM, 2010, pp. 717–719.

[30] A.-K. Pietila¨inen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot, "MobiClique: middleware for mobile social networking," in WOSN '09: Proceedings of the 2nd ACM workshop on Online social networks. New York, NY, USA: ACM, 2009, pp. 49–54.

Kilaparthy Punyavathi Pushpa pursuing M.Tech (CSE) Department of Computer Science Engineering from visakha Institute of Engineering andTechnology,Visakhapatnam.

Panthangi. Pavithra M.tech(Phd) ,working as an Assistant Professor in the Department of Computer Science and Engineering in visakha Institute of Engineering and Technology,Visakhapatnam.