A Decentralized PKI In A Mobile Ecosystem

Varun Chandrasekaran[1, 2] and Lakshminarayanan Subramanian[1]

[1]New York University
[2]University of Wisconsin-Madison

Abstract

A public key infrastructure (PKI) supports the authentication and distribution of public encryption keys, enabling secure communication among other uses. However, PKIs rely heavily on a centralized trusted party called the certificate authority (CA) which acts as the root of trust, and is also a single point of compromise. We describe the design of a decentralized PKI utilizing the intrinsic trust of the cellular (GSM) network. Secure Mobile Identities (SMI) is a repetitive key-exchange protocol that utilizes cryptographic proofs to prove the unique identities of mobile users. In this paper, we discuss how one can create strong reputations for an identity despite the presence of an adversary who can exploit the probabilistic one-way trust property of GSM networks. Our evaluation shows minimal computational overhead on existing devices, and quick bootstrap times of under 10 minutes of mobile interaction despite minimal trust assumptions placed, suggesting easy adoption in today's operational ecosystem.

1 Introduction

Security for services like communication, transactions, etc. are the subject of abundant academic and industrial focus. Most of the aforementioned solutions discussed rely on some form of support to associate identities to the keys used for encryption. Public key infrastructure (PKI) exists solely to support the authentication and distribution of public encryption keys. Without PKI, sensitive information can still be encrypted (ensuring confidentiality) and exchanged, but there would be no assurance of the identity (authentication) of the other party. A crucial weakness in such an infrastructure is its heavy reliability (trust) on a certificate authority (CA) that provides services to authenticate the identities of users. Not only are certificates expensive to purchase, recent compromises expose numerous frailties in such a centralized PKI architecture [1–4, 11, 13, 22]. Given such frailties, an intuitive solution would be to decentralize the PKI where users prove their identities to each other to establish mutual trust - eliminating the need for a CA. We explore this idea in a mobile ecosystem through *Secure Mobile Identities (SMI)* - a protocol to exchange self-certified keys to authenticate unique mobile identities.

Mobile devices today are computationally equipped to perform mathematical operations required by cryptography. They are also pervasive among the world population, and are the ideal springboard to setup the decentralized PKI. To achieve its desired properties (§2.1), the SMI protocol leverages two additional features of the mobile ecosystem, namely: (a) the unique identities offered to mobile users such as their mobile numbers (with country codes), IMSI, IMEI, ICCID[1], or any combination of the aforementioned, and (b) a partially secure communication channel i.e. the *probabilistic one-way trust (POWT) channel* abstraction offered by mobile network operators (MNOs). More precisely, *a mobile device with a unique identity D_i which is genuinely connected to its trustworthy MNO and is weakly authenticated (using GSM authentication) by the MNO; can receive control message M sent by any arbitrary sender, with high probability, provided M successfully reaches the MNO's network.*

The notion of a POWT channel ushers in impediments to forging a strong reputation as the delivery (and consequently security) guarantees are probabilistic in that successful delivery is dependent on the premise that the channel to D_i is not subject to an adversarial presence at the time of delivery. The limited notion of end-to-end trust in the GSM authentication protocol, which is asymmetric in that it does not require the cellular network to identify itself, is fertile ground for adversarial exploits. Mobile devices are exposed to various network-level security vulnerabilities, who can launch different forms of Man-in-the-Middle (MitM) attacks [49]. The recent proliferation in attacks on GSM security [33, 58, 61] has been exacerbated by the widespread availability of software-defined cellular platforms like OpenBTS [16] powered by USRP nodes [21], sysmoBTS [20], Fairwaves [12] and Opencell [17].

However, we assume that such adversaries are economically and computationally constrained, for reasons we allude to in later sections (§2.2). Despite these limitations imposed, a strategically located adversary can masquerade as an honest user through its devices and capabilities by performing eavesdropping, modification, masquerading and phishing etcs, or perform denial-of-service (DoS) - a commonly seen tactic. We employ an

[1]International Mobile Station Equipment Identity, International Mobile Subscriber Identity, and Integrated Circuit Card Identifier

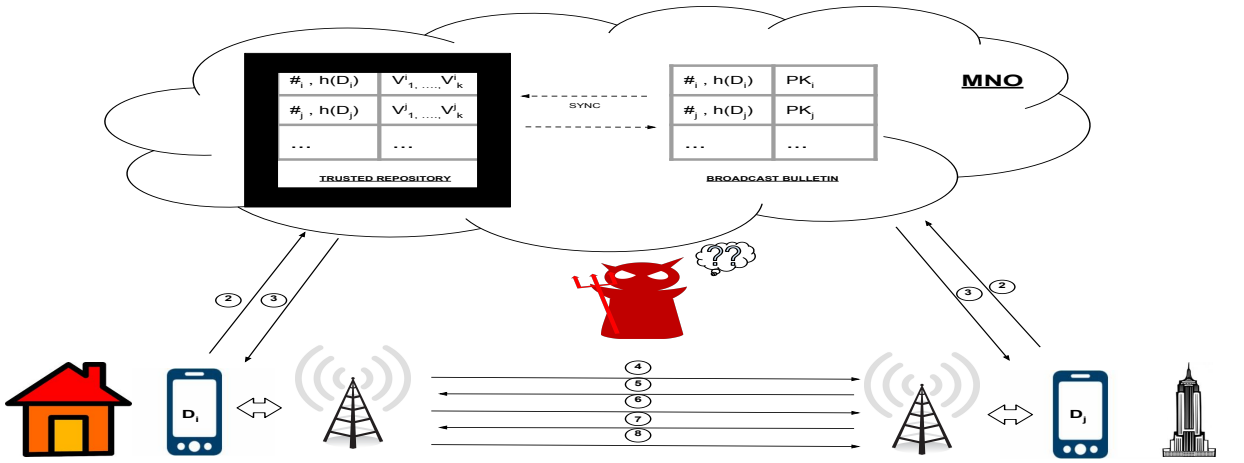

Figure 1—Overview: SMI protocol - refer §4.5 for details

intuitive solution to circumvent such an adversary - mobility. Through a combination of cryptographic proofs and mobility, we ensure that the probability that an adversary succeeds in its efforts are reduced to negligible values; the adversary can also be detected whilst performing malicious actions - which at best delay the completion of the protocol.

To the best of our knowledge, SMI is the first solution focused on building a decentralized PKI through establishing a strong end-to-end trust model around the unique identity of mobile devices. Existing solutions to enhance the security of mobile devices have predominantly focused on either the lower level to secure the wireless channel [26, 44] or the higher level to protect user identities and applications [14, 45, 52], while others have focused on protecting the mobile device itself [36]. Our evaluation suggests that mobile devices do indeed possess the compute power required for our protocol, that the identity-key mapping can be quickly achieved - utilizing 8-10 cryptographic proofs, and the augmentation required to existing infrastructure can be described as minimal at best.

2 Setup and Threat Model

In this section, we define the problem we set out to solve, and establish our threat model. We begin by briefly reviewing current GSM authentication, and how a strategically located adversary can exploit it.

2.1 Problem Definition

In this paper, we wish to design a decentralized PKI by authenticating the unique identities of participating (mobile) users in a distributed fashion. To briefly describe the idea, consider a pair of users i and j, each associated with unique mobile identities D_i and D_j. These devices generate their own public keys PK_i and PK_j and their corresponding private keys. Our goal is to ensure i learns the identity-key mapping (j, PK_j) and likewise for j. With SMI, we aim to achieve the following properties:

1. Secure Identity-Key Mapping. Participating parties are able to prove the uniqueness of their identities (D_i) to each other without leaking any sensitive information.

2. Non-repudiation. Participating parties can prove that they are indeed in communication with each other.

3. Key Ownership. Participating users can prove ownership of the key.

4. Scalable Key Exchange. The protocol scales with the number of participating users.

This ecosystem also includes network adversaries who may try to disrupt the key-exchange process. §2.2 provides some insight to better understand the abilities and limitations of such an adversary in a GSM ecosystem, §2.3 highlights fallacies in the (seemingly) secure LTE ecosystem, and §2.4 formalizes the threat model based on insight from both.

2.2 GSM Security

Background: Subscribers of any MNO initially procure a subscriber identity module (SIM) which has a unique ICCID, IMSI, and a shared key required for authentication. Collectively, we refer to this as User Equipment (UE). The Authentication Center (AuC) of the MNO gains knowledge of this key during SIM registration. The UE and the AuC use an *unidirectional* challenge-response protocol to verify the shared key, and establishes a temporary session key for the subscriber. This (temporary) key is used for securing subsequent communication.

Threats: However, this asymmetric cellular authenti-

cation is vulnerable to eavesdropping and interception, among other threats. One such threat is a masquerade attack - the adversary uses fake base transceiver stations (fBTSs) to trick the device into connecting to an adversarial (henceforth termed fake) network. Such attacks are made more accessible with the availability of software-defined radio platforms that can emulate cellular base station functionality [12, 15–17, 20, 21]. One way to do this involves the fBTSs exhibiting greater signal strength than the real base station to lure mobile subscribers during their connection to the cellular network. Another way is to selectively jam specific frequency bands in a local area to force mobile devices to search for alternative base stations, some of which could be fake.

We now describe the threats in greater detail. Meyer et al. [58] show a man-in-the-middle (MitM) attack on Universal Mobile Telecommunications Systems (UMTS) by exploiting the lack of integrity protection in the base stations. Kostrzewa et al. [51] exploit weaknesses of the A5/2 [33] cipher to demonstrate another MitM attack. The A3 and A8 algorithms used by cellular networks, specifically COMP128 and COMP128-1, are known to have flawed implementations which make it easier for network adversaries to launch MitM attacks against [68]. Another attack is for the fBTS to use known ciphertext attacks [29] to recover the session keys without disrupting the original subscriber authentication process [27]. Given physical access to a SIM card, there have been attacks that can recover the shared key embedded in the SIM through effective challenge-response mechanisms. Extensions of these attacks have facilitated over-the-air (OTA) cracking of shared keys [19, 57].

Cost: Utilizing fBTSs is associated with numerous difficulties, and we highlight a few:

1. It must be ensured that the mobile device of the person under observation is in standby mode and the correct MNO is found out. Otherwise, the mobile device has no need to log in to the simulated base station.
2. Depending on the signal strength of the fBTS, numerous IMSIs can be located. The problem is to find the right one i.e. the one corresponding to the person under observation.
3. All mobile devices in the area covered by the fBTS have no access to the original network, and hence, incoming and outgoing calls cannot be patched through for these subscribers. Only the observed person has an indirect connection.
4. The assignment of an fBTS near the original base station can be difficult, due to the high signal level of the original base station.
5. There are some disclosing factors, such as: (a) a few mobile phones show a small symbol on the display if encryption is not used, (b) since the network access is handled by the fBTS, the receiver cannot see the number of any calling party, and (c) tapped calls are not listed in the bill.

The cost of mitigating the above listed difficulties coupled with the cost associated with the purchase and set up of an fBTS makes using one a challenging (and expensive) proposition.

2.3 3G/4G LTE Security

Though claimed to be more secure, Shaik et al. propose practical attacks against LTE systems [64]. Their work suggests that an attacker can locate an LTE device with reasonably high accuracy. They go on to explain how one can persistently deny some or all services to a target LTE device. As stated by Green [5], hardware is currently programmed to fail over to GSM when the 3G/4G connection is unavailable. Weaknesses in the KASUMI cipher used in 3G have also lead to a different attack [37].

2.4 Adversarial Model

The adversary can utilize an fBTS to monitor and alter any or all communication relayed through it [65], and can also jam specific signal frequencies. Through this form of MitM presence, the adversary can masquerade as a legitimate user in the operational ecosystem. As stated earlier, there is reasonable bootstrapping cost associated with purchasing and setting up the equipment [6]. Hence, we assume that the adversaries are not omnipresent with respect to the locations the user moves to. For practical considerations, we assume that the adversary will neither follow the user to more than a finite number of locations, nor will follow any mobile user continuously. The adversary is computationally bounded and would require some time to cryptographically break the weak authentication layer of GSM [27, 30][2].

3 Overview

In this section, we present an overview of the SMI protocol. We begin by discussing the prerequisites for our protocol, followed by explanation of the various components that constitute it.

3.1 Assumptions

The SMS channel operates over the same control channel used for authenticating mobile subscribers. Given the weak authentication mechanism currently in place, the SMS channel can be viewed as a probabilistic one-way

[2]These attacks are most effective when the adversary is actively on trail of a particular user.

trust (POWT) channel (as defined in §1). The remaining assumptions are extensions of the POWT channel assumption, and are stated below:

1. The cellular service provider (i.e. MNO) and mobile device is trusted. The SMI protocol primarily deals with network-level adversaries who aim to seize the cellular channel to launch MitM attacks.

2. The shared key present in the SIM issued by the MNO is not compromised at the time of issue. If adversaries have the capability to issue their own pre-authenticated SIM cards, then the POWT channel assumption does not hold.

3. If assumptions 1 and 2 hold true, we can assume that SMS messages are delivered with high probability as the internal cellular network is not tapped or tampered, leaving the last hop where it is exposed.

3.2 Protocol Summary

The SMI protocol involves users generating asymmetric key-pairs, and self-certified challenge-responses attesting the ownership of the key. Additionally, the protocol is a combination of two main components, namely (a) zero-knowledge proofs to authenticate unique user identities, and (b) user mobility to circumvent the MitM adversary to incrementally establish trust over the POWT channel. We now elaborate upon these requirements.

Proof of Key Ownership (§3.3): It is insufficient if a party generated an asymmetric key-pair and broadcasts a public key to claim ownership of it. The party has to prove ownership of this key-pair to provide the associated cryptographic security guarantees (such as authenticity of the information shared). Compromised keys need to be revoked, and the keys provided as replacement need to be vetted for ownership as well.

Unique User Credentials (§3.4): Each participating party needs to possess a unique identity to bind to the key it generates. Knowledge of this unique identity helps participating parties authenticate communication across the POWT channel. Thus, we require a globally unique namespace, where there is negligible probability of identity collision, and a computationally inexpensive mechanism to generate unique identifiers.

Circumventing MitM Adversaries (§3.5): For the adversary to achieve success (by disrupting communication), it has to be in the vicinity of the mobile device at all times. Additionally, the adversary is constrained (§2.4) both computationally and economically. A robust protocol design will utilize this knowledge to ensure high evasiveness in the presence of such an adversary.

Zero-Knowledge Proof of Knowledge (§3.6): To state succinctly, a zero-knowledge proof of knowledge (ZKPoK) is a method by which one party (the prover) can prove to another party (the verifier) that a given statement is true, without conveying any information apart from the fact that the statement is indeed true [31]. In the context of SMI, we require a ZKPoK to prove that the unique, non-forgeable identity indeed belongs to a user, without revealing the identity itself. Sharing the identity makes it susceptible to replay attacks [66]. Proof of identity is essential in an adversarial ecosystem as an adversary can no longer masquerade as an honest user, because of its inability to successfully provide a proof of knowledge for identification.

In subsequent sections, we propose techniques to satiate the aforementioned requirements.

3.3 Proof of Key Ownership

For the purposes of easier explanation, let us assume that Alice and Bob are communicating, and Bob is aware of Alice's public key PK_i. Bob wishes to validate the fact that Alice also owns the corresponding secret key SK_i, so as to engage in secure communication (for example). A proof of key ownership can be trivially achieved through the process of encryption. Our assumptions state that the device is secure, hence the private key generated is also secure. Bob begins by encrypting a random value under Alice's public key PK_i, and sharing this with Alice. Alice can convince Bob that she indeed owns the secret key SK_i by responding with the clear text of the random nonce, as decryption is possible only with the corresponding secret key. Proving key ownership requires 2 messages - one to generate the challenge, and one to generate the response.

3.4 Unique User Credentials

A combination of the phone number (with country code), IMSI, IMEI, and ICCID is a unique, non-forgeable identity. Each individual component is a unique namespace, hence its composition is also unique. Additionally, some components are not shared, and kept secret making them hard to forge. Achieving such a unique identity is trivial in the presence of a collision resistant hash function (CRHF) which produces a fixed-length output [7, 63]. A hash function H is collision resistant if it is hard to find two inputs that hash to the same output; i.e., two inputs a and b such that $H(a) = H(b)$, and $a \neq b$. For each user i with $\#_i$ as its phone number (with country code)[3], a unique identity D_i is created as follows:

$$D_i = H(\#_i||IMSI_i||IMEI_i||ICCID_i)$$

where $||$ denotes concatenation.

3.5 Circumventing MitM Adversaries

As stated earlier in §2.4, the adversary in our operational ecosystem is one that can launch various forms of MitM

[3] All other parameters that are input to H are self explanatory

attacks. Though such an adversary is range constrained[4], it is unavoidable while using the SMS channel for its POWT property. Given the various limitations placed on such an adversary, an elegant solution to circumvent is effects is mobility. If users participate in the SMI protocol only whilst they are mobile, they reduce their odds of being in the vicinity of such an adversary, ergo increasing the probability of successful completion. As highlighted earlier in §2.2, the adversary is unable to repeatedly/continually follow a single user, making mobility the ideal solution to thwart such an adversary.

3.6 Zero-Knowledge Proof of Knowledge

We wish to leverage the fact that each user is associated with a unique identity, and further bind this identity with a public key without revealing the identity itself. ZKPoK of identity are well studied, and we use the Fiege-Fiat-Shamir identification scheme in our protocol [40].

Setup: We begin by choosing two large prime integers p and q and computing the product $n = pq$. Both the prover and the verifier receive n, while p and q are kept secret. This is followed by creating secret numbers $s_1, \cdots, s_k$ coprime to n, and verification numbers $v_i \equiv s_i^2 \pmod{n}$. The prover is then sent the secret numbers. The verifier is sent the verification numbers $v_i, \cdots, v_k$ by the prover when he wishes to identify himself to the verifier. The verifier is unable to recover the secret numbers from the verification numbers due to the difficulty in determining a modular square root when the modulus' factorization is unknown.

Procedure:

1. The prover chooses a random integer r, a random sign $s \in \{-1, 1\}$ and computes $x \equiv s \cdot r^2 \pmod{n}$. The prover then sends x to the verifier.

2. The verifier chooses its challenge numbers $a_1, \cdots, a_k$ where a_i equals 0 or 1, and sends this string to the prover.

3. The prover computes $y \equiv r s_1^{a_1} s_2^{a_2} \cdots s_k^{a_k} \pmod{n}$, and sends this response to the verifier.

4. If $y^2 \equiv \pm x v_1^{a_1} v_2^{a_2} \cdots v_k^{a_k} \pmod{n}$, the verifier's query is satisfied.

This procedure requires a total of 4 messages, and is repeated with different r and a_i values until the verifier is satisfied that the prover does indeed possess the modular square roots (s_i) of his v_i numbers.

Security: In the procedure, the prover merely proves to the verifier that he has the secret numbers without revealing what those numbers are. Any eavesdropper would only learn the same information. To break the protocol, the eavesdropper need correctly guess the verifier's a_i, an event of probability 2^{-k}.

[4]fBTSs have a limit on the maximum range within which they operate

Augmentation Required: The SMI protocol requires each user i to upload its verification numbers to a publicly accessible repository such that they are indexed by a digest of its unique identity D_i. Digest generation is typically achieved using one-way functions, preventing any adversary from inverting the digest to obtain the unique identity. The protocol also requires a retrieval function F that takes as input these digests to return the corresponding set of verification numbers. Given user i, with digest $h(D_i)$,

$$F(h(D_i)) = \{v_1^i, \cdots, v_k^i\}$$

where h is a hash function such as SHA-256, and $\{v_1^i, \cdots, v_k^i\}$ represents the set of verification numbers corresponding i's secret numbers. The repository should be tamper-resistant, so as to ensure the integrity of the content it houses.

4 Key Exchange Protocol

In this section, we explain the core protocol used to build credibility of the identity-key pair.

4.1 Secret Generation and Registration

Remember that the Fiege-Fiat-Shammir identification scheme requires the existence of the product of two large primes, n. In our ecosystem, this number is generated and propagated by the trusted MNO. Only the MNO knows the individual primes p and q, making factoring n hard for all other parties. At the time of first connect, each participating party i begins by generating its unique identity D_i and registers its digest $h(D_i)$ with the publicly visible trusted repository. We assume such a registry is under the supervision of the MNO. It then proceeds to generate its secret numbers, and registers the corresponding verification numbers indexed by this digest. Modification of the retrieval function F is of great benefit to the adversary - the adversary can ensure that participants receive the wrong verification numbers, resulting in the failure of the ZKPoK. Hence, we assume that the function is also trusted (i.e. under the supervision of the MNO).

4.2 Strawman Protocol

We begin by discussing the key-exchange strawman in the absence of any adversaries. This will provide readers with insight to protocol behavior to understand how an adversary can impede it. After the steps described in §4.1, the protocol begins with each party i generating their own key-pair $\{PK_i, SK_i\}$. Note that key generation can happen through a variety of schemes, including the usage of elliptic curves. The public key PK_i is registered as a triplet $< \#_i, h(D_i), PK_i >$ upon a broadcast bulletin. We discuss the registration process in §4.4. The triplet proposes a binding between the publicly visible component of the unique identity - the phone number

$\#_i$, the unique identity itself $h(D_i)$, and the public key. Another party j can look up party i's public key through knowledge of $\#_i$, and vice versa.

Before we discuss the steps in the actual protocol, we assume that communicating parties i and j are aware of the other's public key bindings. The protocol itself is a combination of two main components: (a) proof of key ownership, and (b) proof of identity. Proof of key ownership can be achieved through verifying the encryption of random nonces (as stated in §3.3). Operations in asymmetric cryptography are slow, and this is a limitation we must persist through. In our protocol, we utilize the secret key in the key pair as a signing key for a digital signature algorithm (DSA), where signatures made with a secret key can only be verified with the corresponding public key. Digital signatures also provide the property of ensuring message integrity i.e. ensuring that a message is not modified in transit. An obvious question arises - why not encrypt the message using asymmetric cryptography? The binding between a key and an identity is not established. Proof of identity is achieved using the Fiege-Fiat-Shamir identification scheme (as discussed in §3.6). The primary advantage from this scheme is that it is a parallel zero-knowledge proof, where parallel verification limits the interaction between the prover and the verifier. The proof can also be neatly composed such that both parties can behave as the prover and the verifier. We present the protocol below:

1. $i \rightarrow j : x_i, f_{SK_i}(x_i)$
2. $j \rightarrow i : x_j, \{a_1^j, \cdots, a_k^j\}, f_{SK_j}(x_j || \{a_1^j, \cdots, a_k^j\})$
3. $i \rightarrow j : y_i, \{a_1^i, \cdots, a_k^i\}, f_{SK_i}(y_i || \{a_1^i, \cdots, a_k^i\})$
4. $j \rightarrow i : \phi_j, y_j, f_{SK_j}(\phi_j || y_j)$
5. $i \rightarrow j : \phi_i, f_{SK_i}(\phi_i)$

As stated earlier, both parties are aware of the other's public key and unique identity digest $h(D_*)$. Using this digest, parties are able to obtain the corresponding verification numbers with access to trusted function F. The protocol begins with i attempting to prove its identity to j. It does so by computing x_i (as discussed in §3.6), and sharing it with party j along with a signature of x_i. We represent a digital signature as $f_{SK_*}(.)$, where the signing key is SK_* (the secret key of $*$). Here, i is the prover and j is the verifier. As stated earlier, the signature serves as a proof of key ownership, and also provides the dual property of ensuring message integrity. j responds with its challenge string of length k. j reverses its role and behaves as the prover by generating x_j, and the corresponding signature. Steps 4 and 5 contain responses (ϕ_*), which indicate if the identity proof has been accepted or not. Note that SMI is an asynchronous protocol - eliminating overheads induced due to time-based synchronization.

4.3 An Adversarial Setting

Though zero-knowledge proofs are probabilistic proofs, the steps discussed in §4.2 are sufficient to cement the binding between an identity and the corresponding key - a precursor in establishing a decentralized PKI. However, the presence of a truly determined[5] MitM adversary introduces additional complexity. Such an adversary can:

1. Through its devices, masquerade as a legitimate participant.

2. Modify (alter) ongoing communication between honest participants.

3. Drop communication between honest participants.

In this subsection, we discuss how each of the aforementioned attack vectors can be thwarted, and use the insight gained to further optimize the SMI protocol.

Masquerading (I): An adversary can pretend to own a legitimate user $*$'s phone number $\#_*$, or claim that their public key PK_* is its own. It can even claim ownership of the digest $h(D_*)$. However, messages delivered to $\#_*$ are intercepted by the adversary only when the adversary is in the vicinity of $*$, a function of the POWT channel assumption. Once $*$ moves, such forms of ownership will fail. Similarly, pretending to own PK_* and $h(D_*)$ is also thwarted as the adversary can't prove ownership of SK_* or ownership of the secret numbers associated with $F(h(D_*)) = \{v_1^*, \cdots, v_k^*\}$. Though a low probability event, the adversary can prove ownership of the secret numbers as zero-knowledge proofs are probabilistic. However, repeating the protocol several times reduces the probability of adversarial success to near zero values.

Masquerading (II): An adversary can pretend to own a legitimate user's phone number, but can propose a different public key PK_A and different digest $h(D_A)$. However, such an adversary can only participate in the key-exchange protocol if it is in the proximity of the honest user it is masquerading as. Once the user moves, the other participant will notice a public key conflict, and is notified of adversarial presence.

Communication Alteration: Any alteration in the message payload can be detected using the corresponding signature. An alteration in the signature itself voids the entire message. Violating the structure of the communication indicates the presence of an adversary and the protocol can be halted.

Denial of Service (DoS): This scenario is no worse than

[5]An adversary who is able to make progress despite the limitations imposed in §2.4.

one where a network adversary can drop certificates issued by a CA. Fortunately in our ecosystem, such a form of DoS is only feasible when the adversary is in the proximity of a legitimate user. To circumvent such DoS, we initiate (and complete) key-exchanges only when participants are mobile.

Insight Gained:

1. Mobility is essential in both detecting and avoiding a MitM adversary.

2. Since zero-knowledge proofs are probabilistic, repetition (with a different challenge string $a_1, \cdots, a_k$) is one technique that can be used to reduce the soundness error to negligibly small values.

4.4 Broadcast Bulletin

Registration assumes the principle of trust on first use. We assume that on purchasing a SIM card, a user behaves in an honest manner and does not share any of its associated sensitive credentials. In such a scenario, at the time of first connect, the MNO can compute the MSISDN for user i and hash it to obtain a string l_i that it shares with i. As before, let us begin our discussion assuming the absence of an adversary. In such a scenario, any user i can easily add their triplet $< \#_i, h(D_i), PK_i >$ to the broadcast bulletin using the POWT channel. The scenario becomes more complex in the presence of an adversary M who can arbitrarily modify the key listed on the bulletin board i.e. modify PK_i to PK_M. To mitigate the effects of such an adversary, the bulletin board b independently checks if i does indeed possess the corresponding secret key to the public key listed.

1. $i \rightarrow b : f_{SK_i}(l_i)$

2. $b \rightarrow i : \phi_i^b$

In a scenario above, the bulletin b observes a malformed signature as its listing contains PK_M, not PK_i. Additionally, a MitM adversary can intercept messages from the user and attempt to modify it. However, the adversary does not know l_i and hence the check fails. The user is notified (ϕ_i^b) and can rollback the change made to his key. It is essential for the user to check the validity of the listing made on the bulletin board. To do so, the user is required to be mobile (as alluded in §3.5). This procedure also ensures the liveness of an identity. Note that if an adversary's objective is to masquerade as an honest user, it will not modify a user's phone number $\#_*$ or identity digest $h(D_*)$. Modifying $h(D_*)$ can be easily avoided by comparing the value with that listed in the trusted registry (§4.1). Modifying the phone number associated with a key is just another form of DoS.

4.5 Modified Key-Exchange Protocol

Based on insight gained in the previous sections, we now propose a modified key-exchange protocol. The fulcrum of the new protocol are the same steps discussed in §4.2. We collectively call the 5 steps as an interaction. There are, however, 2 key modifications to the protocol. The first is *bulletin verification* (§4.4), where participating users verify if the listing in the bulletin board is indeed accurate. This precludes the second modification which is *repetition of interactions 'n' times*. Doing so reduces the soundness error, consequently reducing the probability of adversarial success. Additionally, each interaction occurs only when the user is mobile, and is initiated by the user himself. If the user does so in a random manner, the adversary would have to: (a) follow the user at all times, and (b) monitor all his communication - both of which are clear violations of its capabilities. An orthogonal technique that can be used to strengthen the security guarantees offered by the zero-knowledge proof is to increase the length of the challenge string, k. The entire protocol is stated below, for clarity.

Handshaking

① $i \rightarrow j$:?!

Bulletin Verification

② $i, j \rightarrow b : f_{SK_i}(l_i), f_{SK_j}(l_j)$

③ $b \rightarrow i, j : \phi_i^b, \phi_j^b$

Interaction (repeated 'n' times)

④ $i \rightarrow j : x_i, f_{SK_i}(x_i)$

⑤ $j \rightarrow i : x_j, \{a_1^j, \cdots, a_k^j\}, f_{SK_j}(x_j || \{a_1^j, \cdots, a_k^j\})$

⑥ $i \rightarrow j : y_i, \{a_1^i, \cdots, a_k^i\}, f_{SK_i}(y_i || \{a_1^i, \cdots, a_k^i\})$

⑦ $j \rightarrow i : \phi_j, y_j, f_{SK_j}(\phi_j || y_j)$

⑧ $i \rightarrow j : \phi_i, f_{SK_i}(\phi_i)$

The handshaking phase listed above is to notify the other participant of protocol commencement. It can be any arbitrary ping (?!) message.

4.6 Using Multiple Channels

Another simple, but extremely useful variant of the key exchange protocol is to leverage multiple communication channels: SMS, data channel, WiFi, Bluetooth etc. To partially alleviate load on the SMS channel, we can use alternative channels to offload the signature portions of our scheme. One cannot completely bypass the use of the SMS channel, since that is the only channel that is directly connected to the identity of the mobile device participating in the protocol, and is crucial for the POWT channel assumption to hold.

DDoS Alleviation: SMI messages coupled with existing text message traffic can increase the network load on the cellular network. Enck et al. [38] suggest that to disturb cellular service, efficiently blanketing only a specific area with messages is sufficient so as to increase the probability of successful disturbance. Similar to the design of backoff protocols in wireless networks, SMI can use increasing delays in SMS messages and message losses as early signals to decrease the rate of messages from individual devices to reduce the network load. Distributing the messages across the SMS and alternate channels will further reduce cellular load.

5 Operational Considerations

In this section, we discuss the operational considerations and improvements to the SMI protocol to ensure easier adoption. Specifically, we discuss: (a) the trade-offs between interactivity and non-interactivity in the context of zero-knowledge proofs (§5.1), (b) scheduling required to optimize message costs (§5.2), (c) colluding adversaries (§5.3), and (d) key revocation (§5.5).

5.1 Non-Interactive Zero Knowledge

Non-interactive zero-knowledge (NIZK) proofs are a variant of zero-knowledge proofs in which no interaction is necessary between prover and verifier. Under the random oracle model, NIZKPoK can be obtained using the Fiat-Shamir heuristic [41]. To better understand, consider the Fiege-Fiat-Shamir proof of identification between a prover p and a verifier v.

1. $p \rightarrow v : x$
2. $v \rightarrow p : \{a_1, \cdots, a_k\}$
3. $p \rightarrow v : y$
4. $v \rightarrow p : \phi$

A non-interactive variant of the same proof requires: (a) the original interactive proof to have the property of being public-coin, and (b) the prover to have access to a cryptographic hash function (say H). H is likened to the random oracle - which would respond to every unique query with a truly random response (however many times). To state the NIZK variant of the above proof:

$p : H(r, s, n) \rightarrow \{a_1, \cdots, a_k\}$
1. $p \rightarrow v : (x, \{a_1, \cdots, a_k\}, y)$
2. $v \rightarrow p : \phi$

The above proof i.e. $(x, \{a_1, \cdots, a_k\}, y)$ avoids interaction as access to the hash function provides the challenge string. Any verifier can verify this proof. The major advantage in using NIZK is the decrease in communication bandwidth - an interaction of length 5 is now reduced to 3.

Security: As stated earlier, the Fiat-Shamir heuristic is proven secure under the assumption of the existence of a random oracle. However, work by Goldwasser et al. [43] disproves this in the standard model. For the purposes of the SMI protocol, the output of hash functions possess sufficient randomness in comparison to a random oracle, and can be used to generate the NIZKPoK. Alternate constructions such as the one proposed by Fischlin [42] can be used.

Replay: In the interactive setting, replays can be avoided as the proof is a function of the random challenge generated by the verifier. In a non-interactive setting, such guarantees can not be provided. Thwarting replay attacks typically require the existence of a unique session identifier. In the absence of a pair-wise common reference string (CRS) between interacting parties, such identifiers too can be forged in our adversarial model. Setting up such a CRS involves making additional trust assumptions, and hence we look into the application of NIZKPoK in future work.

5.2 Management & Scheduling

The number of messages that a mobile device can send is restricted because of cost factors and the network load. Each device leverages the multiple channels available at its disposal to reduce message overhead by transmitting non-critical information using alternative channels which are less constrained (in terms of content size) than the SMS channel (§4.6). Though the current variant of the protocol is user initiated, one can envision a scenario where the protocol automatically commences upon detecting periods of considerable mobility (using information from the phone's other sensors such as accelerators etc.). In such a scenario, the user can pre-specify an ordered list of his contacts whose identity-key mappings he requires. Given a limited message quota for a time period (characterized by costs), the schedule optimization mechanism orders messages based on the highest utility metric of user priority i.e. important users are to be bootstrapped faster.

5.3 Colluding Adversaries

Thus far, we have assumed adversaries in our ecosystem operate in an independent manner. Adversaries that collude can behave in a strategic manner so as to continually inhibit the success of the SMI protocol, either through DoS or through masquerading. As stated earlier, though DoS-based attacks do not affect the security properties

of the protocol, they do hamper normal functioning. Colluding adversaries possess a threat similar to that of a nation state adversary, and our protocol is not equipped to handle such an adversarial model.

5.4 Key Generation

The techniques described in earlier sections work independent of the nature of the keys. Table 1 contrasts the security offered as a function of a varying key lengths (where larger keys offer greater security). It can be observed that providing strong securities comes with a higher cost in the case of standard asymmetric keys, which get progressively larger. Operations with such large keys are compute intensive and slow. Hence, we favor elliptic curve [50] algorithms in production, as the key sizes are entirely practical. We stress that the techniques discussed in the paper work independent of the nature of the algorithms used.

Symmetric	Asymmetric	Elliptic Curve
80	1024	160
112	2048	224
128	3072	256
192	7860	384
256	15360	512

Table 1—Key Length (in bits)

5.5 Key Revocation

In the event of a secret key compromise, a user i revokes the public key currently present in the bulletin board. One can envision an adversary using such a revocation signal (σ) to instantiate yet another form of DoS attack. Luckily, this is avoided because of the existence of the CRS l_i between the user and the bulletin b.

1. $i \rightarrow b : f_{SK_i}(l_i, \sigma)$

2. $b \rightarrow i : \phi_i^b$

$b, i : l_i^{'} \leftarrow m(l_i)$

3. $i \rightarrow b : PK_i^{'}, f_{SK_i^{'}}(l_i^{'})$

4. $b \rightarrow i : \phi_i^b$

Here, user i indicates his desire to revoke the public key associated with secret key SK_i. Upon doing so, both the bulletin and the user modify the CRS using a well-defined transformation to obtain $l_i^{'}$. This is done as the CRS l_i was implicitly bound to the pair $\{PK_i, SK_i\}$. User i can now upload a new public key $PK_i^{'}$, and prove ownership of this key with a signature of $l_i^{'}$ signed with $SK_i^{'}$. Note that the revocation signature can be used only once; this is implicit as revocation is followed by CRS transformation.

5.6 Extending To Multiple Participants

Thus far, we have discussed a variant of the protocol where the identity-key credibility is established in a pairwise fashion. For N users in an ecosystem, establishing pairwise credibility would require $O(N^2)$ executions of the protocol. This highlights the difficulty in a web of trust based scheme, where there is the absence of a central controller (i.e. the CA). New participants, though trusted by some users, will not likely be readily trusted by others until they have interacted. In practice, we expect each participant to interact with only $O(c)$ users, reducing the total number of interactions to $O(Nc)$ i.e. linear in the number of participants.

Multiple Broadcast Bulletins: A single broadcast bulletin is easy to reason about. However, such a design becomes the single point of failure. Multiple broadcast bulletins ensures greater protocol scalability, but introduces synchronization problems i.e. ensuring consistency. The consistency-availability debate is one that is well studied [32], as is replication in a distributed setting, and appropriate consensus and consistency protocols can be employed as required. A situation of particular interest is one where an adversary modifies the key PK_i of a particular user i in a subset of bulletin boards it is published in. Reconciliation based on a quorum fails when majority of the boards are compromised. In such a scenario, the user i is contacted directly and asked to resolve the conflict (refer §4.4).

6 Evaluation

Thus far, we have discussed various components of the SMI protocol. In this section, we evaluate the security of the protocol (§6.1), perform micro-benchmarks (§6.2) to provide insight into real world performance, and use this insight in checking protocol feasibility (§6.3).

6.1 Security Guarantees

To understand the security guarantees, let us first revisit the capabilities (and constraints) of the adversary in our ecosystem. Any adversary that aims to successfully participate in the protocol, and subsume the forged identity, has to be in continuous proximity of the mobile device and completely prevent any communication between the device and the cellular network. Note that the adversary also has to operate in the *always on* mode, to immediately detect and hinder any user activity. This kind of adversarial behavior is severe, and difficult (as highlighted in §2.2). The easier strategy for the adversary is to perform DoS, which may at best, delay protocol convergence.

At any location, let us assume that a user is in the presence of such an adversary with probability p. Assuming no collusion between various adversaries, the probability that the user's communication is disrupted across n locations the user *continuously* visits is p^n. In the average case ($p = 0.5$), the value of p^n is much lower than 1 for larger values of n i.e. an adversary can be present at all locations a user visits with very low probability[6]. Figure 6.1 plots the number of locations (or interactions) required to achieve a desired level of security (scaled up by a factor 100), as a function of adversarial interference. In this context, we define security as the ability to interact without adversarial interference across any (of the total of n) interaction(s) i.e. this is the value of $1 - p^n$.

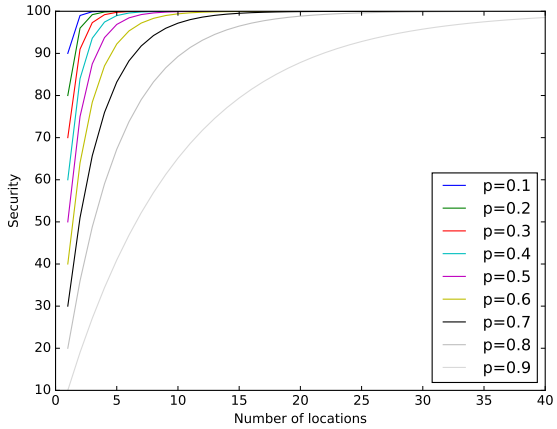

Figure 2—Number of interactions required to obtain security - the higher the security, the better

As alluded in §3.6, the soundness error for the ZKPoK, when repeated n times is 2^{-kn}. That is, the probability of an adversary forging a proof decreases exponentially with an increase in n. From Figure 6.1, we observe that values of $n = 8 - 10$ provide an average level of security of 85, and provide a level of security of 99 for $p = 0.5$. Thus, for (say) 10 repetitions, the soundness error can also be reduced to 2^{-10k}.

6.2 Micro-Benchmarks

We conducted simple micro-benchmark tests on the message length, delivery delays, and energy consumption across a real-world mobile networks in both UAE and the USA. The key observations are as follows:

1. Message Length: The protocol is carefully designed such that the content of each step fits within a single message (140 bytes or 1120 bits). This avoids issues such as message fragmentation (and the related re-assembly delay). Message length can be further reduced using standard loss-less compression techniques to ensure message precision is not lost. Assuming a key size of 384 bits, table 2 discusses the number of messages required. Note that for an key of size $2t$, the signature is of size $4t$. An increase in key size will increase the number of messages by a factor 2 (due to the size of the signature)[7].

Phase	Number of Messages
Posting on Bulletin	2
Bulletin Verification	2
Interaction	5
Revocation	4

Table 2—Messages required

As explained earlier, information pertaining to proof of key ownership i.e. signatures can be shared using alternate channels, decreasing the load shared through the SMS channel.

2. Delivery Delay: Delay in delivery is proportional to the length of the message. It was observed that the delay is 7.4 seconds using T-Mobile in the USA and 7.1 seconds using Etisalat in the UAE for a message of size 140 bytes and scales linearly with the increase in message size. Note that in our case, there is no message fragmentation and the delay can be kept constant. The deviation between the two countries is due to a combination of factors including the allocation of network infrastructure in the specific city of testing, fairness from service providers, coverage and network capacity.

3. Energy Consumption: We used a Google Nexus 5 device (2300 mAh power supply) running Android v5.1.1 (Lollipop) to conduct key exchanges using just the SMS channel, with an aggressive periodicity of 5 minutes. This was done whilst default mobile applications were also running. We observed that the energy consumption was moderate; the SMI protocol execution resulted in roughly a 30% decrease in battery (from full charge) after a 5.2 hour execution of the protocol. In practice, due to the use of multiple channels to achieve a much lower frequency of exchanges, the energy consumption will be much lower. We also believe the energy consumption will further decrease due to recent advancements in the design of new battery technologies [70], and Android upgrades [9, 18].

4. Challenge String Length: As alluded to in earlier sections, the security of the PoK depends upon the length of the challenge string k. However, this value also has implications on the computational overhead imposed on the device (in terms of computing the value of y), and has direct implications on the number of messages required for the protocol. The payload size in a message (352 bits for a key of size 384 bits) and computational capacity of cellular devices (§6.3) allow for flexibility in choosing k.

[6]This value of p is chosen for reasons alluded to in the previous paragraph and in §2.2

[7]We define payload size as $1120 - 4t(mod 1120)$

We suggest a value of around 80, reducing the soundness error to 2^{-800} (assuming $n = 10$)[8]

Insight Gained:

1. A user needs to be mobile for an average time of 60 seconds to successfully participate in one interaction (assuming mobility ensures zero adversarial interference). This time includes that taken for bulletin verification, and any computational overhead associated with the protocol steps.

2. The signatures generated are of constant size, and hence increasing the length (in bits) of the challenge string has no adverse impact on the number of messages required for the protocol.

6.3 Feasibility Study

In this subsection, we perform an additional set of benchmarks geared at measuring the ease of adoption of the protocol, based on insight from §6.2.

Computational Overhead: To understand computational overheads imposed due to standard cryptographic operations, we used the *OpenSSL Speed* benchmarking suite on a processor of clock frequency 800 MHz running no other workloads. Though not conclusive, this evaluation does provide insight into the functionality of a protocol on a mobile device (which is compute constrained), running other workloads.

Table 3 highlights the efficiency of the ECDSA algorithm, while Table 4 indicates the ease with which hashing can be achieved.

Size	Sign	Verify	Sign/s	Verify/s
160 bit	0.0006s	0.0023s	1682.8	438.7
192 bit	0.0011s	0.0041s	876.5	243.1
224 bit	0.0010s	0.0020s	997.9	495.0
256 bit	0.0012s	0.0033s	812.8	304.5
384 bit	0.0021s	0.0118s	475.2	84.5

Table 3—Signature Overhead

Type	16 bytes	64 bytes
SHA-256	4371.17k	7487.96k
SHA-512	3225.67k	10109.53k

Table 4—Hashing Overhead, values in cells are in 1000s of bytes per second processed

Storage Overhead: In-device storage is primarily due to the keys and the PoK. However, both are negligible in size and induce close to no overhead. The size of both the trusted repository (§4.1) and bulletin board (§4.4) grows linearly in the number of participating users. In the worst case (assuming a storage of 500 B/user), we estimate that the storage is nominal - 500 GB for one billion users. Both these boards can be implemented using a hash-table for constant time lookups.

Mobility Required: Thwarting MitM adversaries necessitates user mobility. An immobile user is at risk of being in the vicinity of such an adversary, and can never complete the protocol. This is one of the serious drawbacks of our protocol design, one that is unavoidable because of our reliance on the POWT channel.

Insight from §6.2 states that the SMI protocol requires an average time of 60 seconds for a single interaction[9]. We also require $8 - 10$ interactions to eclipse the effects of a MitM adversary (§6.1). Thus, this evaluation is geared at checking the degree of human mobility. To this end, we utilize a location data-set from CRAWDAD [62]. The data-set compromises of human mobility in 5 different settings (M1,.., M5). Table 5 provides more information about these settings.

Abbrv.	Location	# of Participants
M1	KAIST	92
M2	NCSU	35
M3	New York	39
M4	Orlando	41
M5	Fair	19

Table 5—Mobility Survey Overview

In specific, the data-set contained location (x, y) coordinates of the participants. We define mobility of participant i, μ_i, as the fraction of times he moves a (Euclidian) distance greater than Δ meters in a time-period of δ minutes. Table 6 is populated with the average value of mobility across all participants i.e. $\sum_i^N \mu_i/N$.

Δ	δ	M1	M2	M3	M4	M5
200	1	0.027	0.034	0.084	0.027	0.006
	5	0.113	0.094	0.258	0.105	0.13
	10	0.15	0.127	0.314	0.208	0.275
400	1	0.01	0.019	0.035	0.016	0.003
	5	0.056	0.067	0.205	0.046	0.018
	10	0.10	0.097	0.261	0.084	0.077
600	1	0.004	0.011	0.019	0.012	0.002
	5	0.034	0.043	0.13	0.033	0.010
	10	0.07	0.084	0.233	0.059	0.021
800	1	0.002	0.006	0.013	0.009	0
	5	0.027	0.037	0.106	0.030	0.001
	10	0.049	0.069	0.209	0.051	0

Table 6—Average Mobility

Though not ideal in most cases, M3 - a city ecosystem - does show promising results. We stress that these results

[8]Note that further increasing the k value is still possible

[9]Assuming round-trip latency of 15-16 seconds.

are not definitive - as the participants are not a true representation of the global demographic - but only serves as a first-order metric to suggest that human mobility is non-zero, and is a viable weapon against our MitM adversary. We also notice an increase in mobility with an increase in the δ interval, providing some insight into selecting the right δ interval for interaction repetition. That is, interactions should be spaced out by $O(r\delta^*)$ minutes - where $\delta^*(= 10)$ is an optimal, empirical value for separation, and $r \in [1, 2]$.

7 Related Work

Authentication Fallacies: Toorani and Beheshti [68] outline many of the weaknesses in the GSM authentication layer and propose simple fixes. The GSM specifications team [24, 47] proposed new cryptographic mechanisms to improve upon the early algorithmic vulnerabilities that plagued the original 2G authentication design. However, even the UMTS extensions had several security problems [59]. The LTE authentication mechanism [28] builds upon the UMTS security model and introduces an key derivation hierarchy to enhance the security of the pre-shared keys; even these extensions suffered from several security problems [28, 46, 69]. Han and Choi [46] demonstrate a threat against the LTE handover key management. Tsay et al. [69] find an attack on the UMTS and LTE AKA protocols using an automated protocol analyzer based on a computational model. Tang et al. [67] provide a detailed analysis of the security properties and vulnerabilities of different mobile authentication mechanisms.

Device Pairing: Device pairing or key setup between two devices has been extensively studied [10, 25, 34, 48, 54]. SDDR [53] provides secure encounters whilst enabling secure communication, providing selective linkability and silent revocation. SMILE [56] establishes trust between individuals who have shared a provable encounter. At the site of the encounter, MeetUp [60] proposes a visual authentication scheme using a trusted authority which attests a users public key to its picture. Secure Location Sharing (SLS) by Adams et al. [23], like SMI, uses multiple communication channels along with contextual question/answer protocols to prevent MitM attacks during device pairing. Tamper-evident pairing (TEP) [44] is a protocol that provides simple, secure WiFi pairing and protects against MitM attacks without an out-of-band channel. GAnGS [35] exchanges the public keys of group (of constrained size) members such that each member obtains the authentic public key of the other - at the expense of continuous physical proximity. SPATE [55] achieves a similar goal, aided by visual channels and physical interactions. Both GAnGS and SPATE enable bystanders to learn contact information, disclosing potentially sensitive private information. SafeSlinger [39], provides a system that leverages prior physical encounters to establish trust.

8 Discussion

Protocol convergence can be accelerated by a factor 2 if NIZK were used instead of the standard, interactive variant. However, as stated earlier, NIZK is limited by its ability to be replayed. Replay-based attacks can be thwarted with the presence of a CRS between interacting parties, requiring additional trust assumptions. We wish to explore designs where such CRSs can be generated using alternate trust channels in future work.

Though this paper describes a framework towards establishing a decentralized PKI, it doesn't describe the building blocks of the PKI itself - such as key discovery protocols, or (trusted) infrastructural support for hosting the identity-key mapping. In comparison to other decentralized protocols, SMI holds an advantage in that it is not constrained by the physical location of participants, nor does it impose any additional cost upon participants for credential verification (as it doesn't require them to meet in person).

We assume users possess the ability to use the SMS (POWT) channel an arbitrary number of times, with minimal or no expenditure. While this is the case in countries such as the USA, it isn't in others such as India. We believe that with support from the MNO, our scheme can easily be adopted in the emerging markets as well.

By relaxing our minimal trust assumptions, we can further decrease the $O(cN)$ execution bound. This can be achieved by introducing a hierarchical-trust based model, which will further reduce the constant factor c; for example, introducing the presence of trusted third-party interaction sites with whom users can prove their identity-key mapping using a variety of channels to offset the number of interactions. Another scheme can involve multiple bulletin boards with hierarchical trust; bulletin boards can themselves participate in the protocol, and users interacting with a bulletin board of higher trust are validated faster. We leave this to future work.

As noted as a recurring theme in the paper, the biggest impediment to our system is an adversary performing DoS. Detecting DoS in a preemptive fashion is a challenging, unsolved research problem. However, we can use various signals [8] to detect such an adversary earlier, and respond quicker. We leave these details to future work as well.

Finally, we apologize for the minification of URLs in the references. We assure readers that these are valid URLs, and were only minified for the purposes of aesthetics.

References

[1] *http://bit.ly/2hYXbfq*.

[2] *http://bit.ly/2iYCIcq*.

[3] *http://bit.ly/2jc6TfM*.

[4] *http://bit.ly/1Hlskkq*.

[5] *http://bit.ly/2cSLFoN*.

[6] *https://srlabs.de/bites/decrypting-gsm/*.

[7] *http://bit.ly/2cpYMvC*.

[8] Android-imsi-catcher-detector. *http://bit.ly/2hNF38h*.

[9] Android issues. *https://www.androidpit.com/how-marshmallow-affects-the-nexus-5-battery-life*.

[10] Bump. *http://bu.mp/*.

[11] Diginotar compromise. *http://bit.ly/2jcbCOz*.

[12] Fairwaves. *https://fairwaves.co/wp/equipment/*.

[13] Google compromise. *http://bit.ly/2hYXqXT*.

[14] Lookout. *https://www.lookout.com/*.

[15] Openbsc. *http://bit.ly/2d8Z50A*.

[16] Openbts. *http://openbts.org/*.

[17] Opencell. *http://www.rangenetworks.com/*.

[18] Optimisations. *https://www.androidpit.com/nexus-5-battery-tips*.

[19] Ota sim attacks. *https://srlabs.de/rooting-sim-cards/*.

[20] Sysmobts. *http://www.sysmocom.de/products/sysmobts*.

[21] Usrp. *http://www.ettus.com/*.

[22] Wosign compromise. *http://bit.ly/2iYPx6v*.

[23] A. K. Adams and A. J. Lee. Combining social authentication and untrusted clouds for private location sharing. In *Proceedings of the 18th ACM symposium on Access control models and technologies*, pages 15–24. ACM, 2013.

[24] J. Arkko and H. Haverinen. Extensible authentication protocol method for 3rd generation authentication and key agreement (eap-aka). 2006.

[25] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *NDSS*, 2002.

[26] N. Bambos, S. C. Chen, and G. J. Pottie. Channel access algorithms with active link protection for wireless communication networks with power control. *Networking, IEEE/ACM Transactions on*, 8(5):583–597, 2000.

[27] E. Barkan, E. Biham, and N. Keller. Instant ciphertext-only cryptanalysis of gsm encrypted communication. In *Annual International Cryptology Conference*, pages 600–616. Springer, 2003.

[28] A. N. Bikos and N. Sklavos. Lte/sae security issues on 4g wireless networks. *Security & Privacy, IEEE*, 11(2):55–62, 2013.

[29] A. Biryukov and E. Kushilevitz. From differential cryptanalysis to ciphertext-only attacks. In *Advances in CryptologyCRYPTO'98*, pages 72–88. Springer, 1998.

[30] A. Biryukov, A. Shamir, and D. Wagner. Real time cryptanalysis of a5/1 on a pc. In *International Workshop on Fast Software Encryption*, pages 1–18. Springer, 2000.

[31] M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 103–112. ACM, 1988.

[32] E. Brewer. A certain freedom: thoughts on the cap theorem. In *Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 335–335. ACM, 2010.

[33] M. Briceno, I. Goldberg, and D. Wagner. An implementation of the gsm a3a8 algorithm.(specifically, comp128.), 1998. *Online: http://www. gsm-security. net/papers/a3a8. shtml*.

[34] C. Castelluccia and P. Mutaf. Shake them up!: a movement-based pairing protocol for cpu-constrained devices. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 51–64. ACM, 2005.

[35] 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 *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 92–103. ACM, 2008.

[36] F. S. Cook and D. Satapathy. Method and system for securing a mobile device, Aug. 24 2010. US Patent 7,783,281.

[37] O. Dunkelman, N. Keller, and A. Shamir. A practical-time attack on the a5/3 cryptosystem used in third generation gsm telephony. *IACR Cryptology ePrint Archive*, 2010:13, 2010.

[38] W. Enck, P. Traynor, P. McDaniel, and T. La Porta. Exploiting open functionality in sms-capable cellular networks. In *Proceedings of the 12th ACM conference on Computer and communications security*, pages 393–404. ACM, 2005.

[39] M. Farb, Y.-H. Lin, T. H.-J. Kim, J. McCune, and A. Perrig. Safeslinger: easy-to-use and secure public-key exchange. In *Proceedings of the 19th annual international conference on Mobile computing & networking*, pages 417–428. ACM, 2013.

[40] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of cryptology*, 1(2):77–94, 1988.

[41] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 186–194. Springer, 1986.

[42] M. Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In *Annual International Cryptology Conference*, pages 152–168. Springer, 2005.

[43] S. Goldwasser and Y. T. Kalai. On the (in) security of the fiat-shamir paradigm. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 102–113. IEEE, 2003.

[44] S. Gollakota, N. Ahmed, N. Zeldovich, and D. Katabi. Secure in-band wireless pairing. In *USENIX security symposium*, 2011.

[45] N. Haller, C. Metz, P. Nesser, and M. Straw. A one-time password system. Technical report, RFC 1938, May, 1996.

[46] C.-K. Han and H.-K. Choi. Security analysis of handover key management in 4g lte/sae networks. *Mobile Computing, IEEE Transactions on*, 13(2):457–468, 2014.

[47] H. Haverinen and J. Salowey. Extensible authentication protocol method for global system for mobile communications (gsm) subscriber identity modules (eap-sim). 2006.

[48] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H.-W. Gellersen. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *Ubicomp 2001: Ubiquitous Computing*, pages 116–122. Springer, 2001.

[49] R. P. Jover and P. Giura. How vulnerabilities in wireless networks can enable advanced persistent threats. *International Journal on Information Technology (IREIT)*, 2013.

[50] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987.

[51] A. Kostrzewa. *Development of a man in the middle attack on the GSM Um-Interface*. PhD thesis, Master Thesis, Technische Universität Berlin, 2011.

[52] B. Lakshmiraghavan. Two-factor authentication. In *Pro ASP. NET Web API Security*, pages 319–343. Springer, 2013.

[53] M. Lentz, V. Erdélyi, P. Aditya, E. Shi, P. Druschel, and B. Bhattacharjee. Sddr: Light-weight, secure mobile encounters. In *USENIX Security*, 2014.

[54] J. Lester, B. Hannaford, and G. Borriello. are you with me?–using accelerometers to determine if two devices are carried by the same person. In *Pervasive computing*, pages 33–50. Springer, 2004.

[55] 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 *Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 1–14. ACM, 2009.

[56] J. Manweiler, R. Scudellari, and L. P. Cox. Smile: encounter-based trust for mobile social services. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 246–255. ACM, 2009.

[57] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. Radio-telepathy: extracting a secret key from an unauthenticated wireless channel. In *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 128–139. ACM, 2008.

[58] U. Meyer and S. Wetzel. A man-in-the-middle attack on umts. In *Proceedings of the 3rd ACM workshop on Wireless security*, pages 90–97. ACM, 2004.

[59] M. A. Mobarhan, M. A. Mobarhan, and A. Shahbahrami. Evaluation of security attacks on umts authentication mechanism. *International Journal of Network Security & Its Applications*, 4(4):37–52, 2012.

[60] A. Mohaisen, E. Y. Vasserman, M. Schuchard, D. Foo Kune, and Y. Kim. Secure encounter-based social networks: requirements, challenges, and designs. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 717–719. ACM, 2010.

[61] D. Perez and J. Pico. A practical attack against gprs/edge/umts/hspa mobile data communications. *Black Hat DC*, 2011.

[62] I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong. Crawdad. *http://bit.ly/2d6cQce*.

[63] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[64] A. Shaik, R. Borgaonkar, N. Asokan, V. Niemi, and J.-P. Seifert. Practical attacks against privacy and availability in 4g/lte mobile communication systems. *arXiv preprint arXiv:1510.07563*, 2015.

[65] M. Sthlberg. Radio jamming attacks against two popular mobile networks. 2000.

[66] P. Syverson. A taxonomy of replay attacks [cryptographic protocols]. In *Computer Security Foundations Workshop VII, 1994. CSFW 7. Proceedings*, pages 187–191. IEEE, 1994.

[67] C. Tang, D. Naumann, S. Wetzel, et al. Analysis of authentication and key establishment in inter-generational mobile telephony. In *High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC), 2013 IEEE 10th International Conference on*, pages 1605–1614. IEEE, 2013.

[68] M. Toorani and A. Beheshti. Solutions to the

gsm security weaknesses. In *Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST'08. The Second International Conference on*, pages 576–581. IEEE, 2008.

[69] J.-K. Tsay and S. F. Mjølsnes. A vulnerability in the umts and lte authentication and key agreement protocols. In *Computer Network Security*, pages 65–76. Springer, 2012.

[70] Y. Yang, G. Yu, J. J. Cha, H. Wu, M. Vosgueritchian, Y. Yao, Z. Bao, and Y. Cui. Improving the performance of lithium–sulfur batteries by conductive polymer coating. *Acs Nano*, 5(11):9187–9193.