SPACE: Secure Protocol for Address-Book based Connection Establishment

Ganesh Ananthanarayanan[†,1], Ramarathnam Venkatesan[†,‡,2,3], Prasad Naldurg[†,4], Sean Blagsvedt[†,5] and Adithya Hemakumar[†,5]

[†]Microsoft Research India, [‡]Microsoft Research Redmond
{ganeshan, venkie, prasadn, seanb, t-adith}@microsoft.com

Abstract

We present SPACE an application-level protocol for secure automatic ad-hoc connection-establishment between two devices based on their address book entries. Our protocol is based on the simple premise that if two people have each others contact details in their address books, they probably know and trust each other in some limited way and this can form a basis for a trust relationship between their devices, without additional user intervention. We show how our protocol is resistant to specific security attacks and can accommodate for privacy concerns. Existing connection-establishment protocols for Bluetooth and IEEE 802.11 have known security flaws, and can be compromised using well-known techniques and off-the-shelf hardware. In addition, these protocols require explicit user intervention, like entering a passkey. We believe that these factors have directly impacted the widespread application of ad-hoc networking in the context of mobile phones and other consumer devices.

1 INTRODUCTION

The last decade has seen a huge increase in the number of consumer devices with integrated wireless communications (Wireless Local Area Network WLAN) such as Bluetooth and Wi-Fi that enable users to connect to other nearby devices in a direct peer-to-peer fashion. The potential benefits of these direct connections were once heralded as significant allowing users to easily share resources such as files, computational power and network connectivity, and engage in collaborative applications. With the possible exception of the most recent generation of dedicated portable videogame players, we feel that the potential of peer-to-peer ad-hoc networking has not yet been achieved, especially on mobile phones and laptop computers. We identify two particularly weak points in the connection establishment process viz., security concerns and user inconvenience.

Users are wary about unauthorized and potentially malicious access to their devices that could compromise the privacy of their data. Existing connection establishment mechanisms in Bluetooth and IEEE 802.11x can be compromised using well-known techniques and off-the-shelf hardware [8, 11] and incidents of Bluetooth-borne viruses have been reported. In addition, the connection establishment techniques are cumbersome and often require explicit user intervention. In most Bluetooth implementations, the user must enter a passkey [12] for each connection.

We model trust relationships in such scenarios on real-life relationships among users and devise an automatic and secure application-level protocol for ad hoc connection establishment. Our guiding insight is that if two people have each others contact details (e.g. phone number, email address) in their address books, it means that they are more willing to trust each other in some limited way, and this can form the basis for a trust relationship for their respective devices, without additional user intervention. We believe that if the problem of trusted automatic connection establishment can be satisfactorily solved, this could provide the basis for many other exciting applications such as file and connection sharing and a connected world where our devices are smart enough recognize the people around us that we know, just as humans. Subsequently, additional security and access controls can further improve the security of this connection.

To this end, we present *SPACE*, an application-level protocol for automatic ad hoc connection establishment between two devices based on their address book entries. *SPACE* has two phases. In the first phase - *Scan Phase*, the devices broadcast a keyed, cryptographic hash of the users contact details and also check the presence of the address corresponding to an incoming hash in their address book. In the second phase - *Authentication Phase*, they establish a shared secret key between them via a secure external network channel, which is resilient to impersonation, such as the exchange of SMS or email messages, and subsequently authenticate each other using an encrypted nonce exchange.

We make two important contributions: (1) we propose a protocol for an automatic ad hoc connection establishment between mobile devices based on a novel model of trust that

[1]Hardware, Communications and Systems Group
[2]Cryptography, Security and Algorithms Group
[3]Cryptography and Anti-Piracy Group
[4]Rigorous Software Engineering Group
[5]Advanced Development and Prototyping Group

is founded on contact details, and (2) we solve the problem of impersonation in this setting by performing a key-agreement over an external network that provides a reasonable guarantee of identity pertinent to the specific contact details that are being exchanged (e.g. the key agreement for mobile numbers occurs via SMS). We analyze the security of our protocol and show that it is resistant to impersonation attacks unlike Bluetooth [4, 12] and IEEE 802.11b [4].

2 TRUST MODEL

Devising trust models for ad hoc networks is still an open and challenging area of research. Ad hoc networks can be broadly classified in to two categories based on their method of trust establishment [2, 13]. *Managed* ad hoc networks are based on the assumption of the existence of a central authority to distribute and verify *certificates*. In *Pure* ad hoc networks trust is dynamic and is based on reputation and recommendations from peers in the network. While the models for managed ad hoc networks require a central authority for trust establishment, their counterparts for pure ad hoc networks assume the absence of any initial knowledge vis-a-vis trust. We believe that the assumptions of a central authority and absence of any prior information significantly reduces the utility and effectiveness of peer-to-peer ad hoc connectivity.

We try to model trust in peer-to-peer ad hoc networks based on real-life relationships. Popular forms of peer-to-peer ad hoc connectivity are generally based on a *social* model of trust, e.g. share photos via Bluetooth with devices of people who you know personally. As a natural extension, people are likely to store the contact details (like phone numbers) of people who they know and we leverage this as a basis for trust in *SPACE*. Devices that have each other's contact details in their address books are likely to trust each other, at least in some limited form, and should be able to automatically detect and connect to each other.

Our trust model has the characteristics that are a hybrid of trust models for managed and pure ad hoc networks. While we assume no central authority for trust management, we piggyback on the existing and popular mechanism of exchange of contact details for our *certificate* distribution. The significant advantage of our trust model is its ready deploy ability unlike existing security mechanisms for peer-to-peer ad hoc networks (e.g. there is no viable distribution mechanism for Personal Identification Numbers (PIN) in Bluetooth).

Our trust model raises two relevant questions - (1) people may not trust everyone in their address books, and (2) a user's address book may not contain his entire list of trusted people. The former can be addressed easily by adding an extra field in the address book that specifically marks whether a contact can be trusted for peer-to-peer ad hoc connectivity. The latter problem falls in the category of the larger unsolved problem of *certificate* distribution for peer-to-peer ad hoc networks. We propose a partial solution to this by utilizing real-life relationships for trust establishment.

3 PROTOCOL DESCRIPTION

In this section we describe *SPACE* protocol. Consider two users Alice and Bob with address books $AD_a = \{a_1 ... a_m\}$ and $AD_b = \{b_1 ... b_n\}$ where a_i and b_i represent the individual addresses, and m and n are the sizes of the address books. Let their self-addresses be s_a and s_b respectively. We will refer to the devices as Alice and Bob in the rest of the paper. The protocol has two phases.

3.1 Scan Phase

In *Scan Phase*, Alice and Bob verify the presence of each others contact details in their respective address books. The phase starts with Alice and Bob computing a keyed cryptographic hash $H_{ka}(s_a)$ and $H_{kb}(s_b)$ of their contact details, using randomly generated keys k_a and k_b, and broadcasting it along with the key. Fig 1 represents a part of the broadcast that happens during Scan Phase.

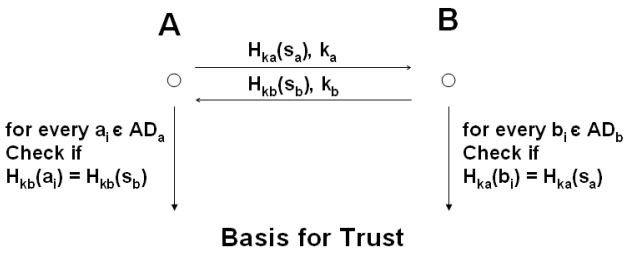

Figure 1. Scan Phase of the *SPACE* protocol

H can be any pre-image resistant and collision-free hash function like SHA_1. Since the hash keys k_a and k_b are randomly and periodically changed, the hash values are different for every broadcast. This prevents the association of the hash value itself as an identity of a device, which may raise privacy concerns.

On receiving this message, Alice and Bob scan their address books to check if the hash of any of the entries in their address books matches the value they have obtained. By assumption the hash value cannot be inverted to get the contact detail in feasible amount of time and hence this guarantees privacy. Also, we can increase the search space by including some additional information with the address before computing the hash value to give a better guarantee of the hash value being non-invertible. But it is to be noted that an adversary can partially invert the hash and obtain a few bits out of it (e.g. obtain the bits corresponding to the area code in the hash and hence violate privacy). While this is not a problem currently with hash functions like SHA_1, we can provide a complete guarantee against inversion by adding a random key to the hash value and re-hashing it.

If Alice and Bob find an address in their address books whose hash matches the value they obtained from the other party, they have a basis for believing that the other person is present in their address book and hence can be trusted. If

Alice or Bob cannot find such an address in their address books, the protocol halts.

3.2 Authentication Phase

In most real-world scenarios, the address books contain either an e-mail address or a phone number. Since the confidentiality of these contact details cannot be guaranteed, this can result in malicious users impersonating their contact details. Consider the scenario where there is a malicious device Ian that wants to connect to Alice. If Ian can find an address $i \in AD_a$, then he can claim his address to be i and hence would get authenticated by Alice because Alice has no means of ascertaining the veracity of Ians claim. This phase deals with the problem of impersonation where the device has no method of verifying the veracity of the address claimed by the other device in the Scan Phase. While Scan Phase identifies the nodes present in a device's WLAN, Authentication Phase ensures that the identities of those nodes are authentic.

We introduce a one-time key agreement step in *SPACE*. Here Alice and Bob agree on a shared key between them via a secure external network channel, which is relatively safe as a way to detect who is sending it - if not as secure as a method for exchange of information such as the exchange of SMS. With devices increasingly becoming multi-homed, we believe that the assumption of an external network channel is reasonable. Fig 2 describes the Elliptic curve based Diffie Hellman key agreement protocol. P is a point on the elliptic curve [3, 7] and is publicly known. Note that Alice and Bob arrive at the same key ($a.b.P = b.a.P$) *without explicitly transmitting the key at any point in time*. The key agreement protocol is based on the hardness of the Elliptic Curve Discrete Logarithm Problem [6] (i.e.) given $a.P$ and P, it is *computationally hard* to find a. Elliptic curve based key agreement was chosen because of its small key size and relatively low overhead in the cryptographic operations [3, 7]. In the first connection instance between two devices, they securely agree to a key between them and use this key for authentication. Note that *SPACE* performs key *agreement* and not key *exchange*, and hence is not susceptible to passive interception of messages.

A point to note is that nodes impersonating their identities (contact details) will not be able to perform the key agreement successfully and will not have the correct key values. This holds true in case of the common address book entries like phone numbers and e-mail addresses. If the key agreement is performed via SMS or mail exchange, then a person impersonating his contact detail will not receive the messages and hence unable to compute the key.

The key is unique for every device pair. Every device has a key table KT with fields contact address c and key k. This table is initially empty and can grow to a maximum length of the size of the address book.

Now Alice and Bob present a nonce-challenge to validate each other. Alice generates a random nonce N_a, encrypts with the key K_{ab}, the key corresponding to the contact address s_b in Alices key table KT_a, and sends it to Bob over the WLAN. Bob uses his key K_{ba} to decrypt the message and obtain N_a. If Alice and Bob had successfully performed their key agreement then $K_{ab} = K_{ba}$. Bob concatenates N_a along with his own nonce N_b, encrypts it with K_{ba} and sends it to Alice. Alice checks the correctness of N_a and sends back N_b for Bobs verification. If Alice and Bob decrypt and send the right nonce values back to each other, they can conclude that there is no impersonation.

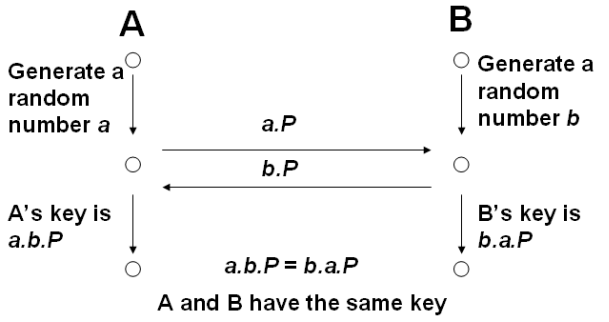

Figure 2. Elliptic Curve Diffie Hellman Key Agreement

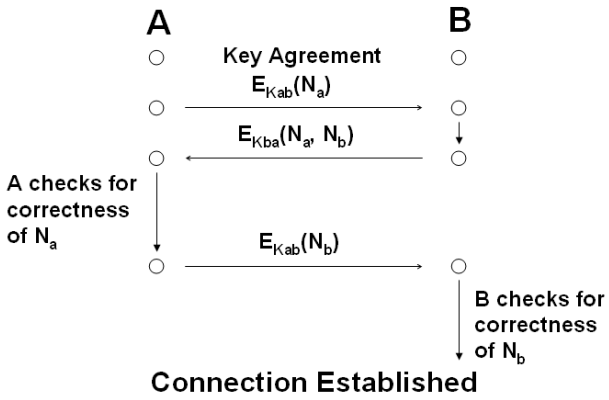

Figure 3. Authentication Phase of the *SPACE* protocol

4 Security Analysis

In this section, we present our analysis of the *SPACE* protocol described in Section 2. We do not present a formal proof of security protocols but instead outline the security assumptions and present our arguments. In the Scan Phase, the preimage resistance of the hash functions is necessary for privacy concerns. In the Authentication Phase, the messages in the key agreement protocol need to be authenticated to be tamper-resistant when necessary and the encryption function used for the nonce challenge needs to be secure. Hence, the security of the protocol primarily depends on the following factors:

1. The security of the standard symmetric key encryption algorithms, such as AES.

2. The keys agreed upon using the public key exchange protocol are indistinguishable from those obtained from a cryptographic pseudorandom number sequence; otherwise an adversary may guess the some bits of the information.

3. Impersonation and message tampering does not happen in the secure channel over which the key agreement happens.

One should choose the parameters such as the key lengths as per the existing standards for public and private key cryptographic primitives.

The final item is the assumption that impersonation and data tampering is not possible in the network over which the key agreement is performed. Passive interception of packets without tampering has no bearing on the security of our protocol (see Section 4.4). In our actual implementation and testing, we used the Short Messaging Service (SMS) over the cellular network for key agreement. The Universal Mobile Telecommunication System (UMTS) has adopted an enhanced authentication and key agreement protocol for 3G communications [1] which includes data confidentiality and user identity privacy [9]. Hence, one may assume that it is sufficiently hard to impersonate and tamper with data in the cellular network.

We now examine some specific attacks which have exposed weaknesses in existing protocols [4, 12], and discuss their impact on our protocol.

4.1 Man-In-the-Middle Attacks

In this attack, a malicious user Ian intrudes into a conversation between Alice and Bob in the WLAN. Ian obtains the messages from Alice and forwards it to Bob and vice versa and makes them believe that they are talking to each other.

In our protocol, in the Scan Phase, Ian obtains the hash of Alices number and forwards it to Bob and does the same with the hash of Bobs number. Hence now Alice and Bob can establish a basis for trust between them even though they are not talking to each other directly.

However in the Authentication Phase, Ian cannot participate in the key agreement protocol and cannot compute the shared key. When he receives an encrypted challenge in the Authentication Phase, he cannot decrypt it to obtain the nonce. In effect Ian can act as a wire in between or scuttle the protocol by tampering with the messages before forwarding it. Therefore, Alice and Bob will not end up connecting to each other. This is futile for Ian and hence this is not a security vulnerability.

Replay Attack: Consider a malicious user Ian who can record and replay the broadcast from Alice to Bob. Bob will be able to find a match in his address book and conclude that there is a basis for believing Ian. Ian can ignore the step of finding a match in his address book. However, because of the Authentication Phase, Ian will not be able to authenticate successfully. If Alice and Bob are connecting for the first time, Ian will not be able to obtain the key agreement messages and will not be able to compute the key. If Bob already has an entry for Alice in his key table, it is infeasible for Ian to compute the key.

4.2 Contact Detail Compromise

In common scenarios, safeguarding ones contact detail (e.g. phone number, e-mail address) is very difficult and is highly likely to be known to people who need not be trustworthy. But this is not a security vulnerability in *SPACE* because it is based on the notion of whether a device has the other devices contact detail in its address book. As long as the device does not engage in a secret key agreement protocol with some user who is malicious (e.g. it trusts that people in its address book are non-malicious), it is secure.

4.3 Denial of Service

A mobile device is especially vulnerable to denial of service attacks as it has limited energy (battery) resources. With respect to our protocol, a determined attacker can effectively mount denial of service attacks. An attacker Ian can impersonate himself to have any address in Bobs contact list AD_b. Hence he will pass through the Scan Phase but will fail in the Authentication Phase.

In the Scan Phase, Bob has a computation cost involved in hashing all the addresses in his address book. This cost can be reduced by precomputing and storing the hashes once and reusing them. Since, we use keyed hashes, Bob can periodically refresh the keys and re-compute the hash values. If Ian impersonates someone for whom Bob already has a key table entry, then Bob will be forced to perform one encryption operation for its nonce and one decryption operation to check for the correctness of the nonce received from Ian. If Ian impersonates someone with whom Bob has never connected before then he would make Bob perform a useless key agreement process which would make him incur computation as well as communication costs. The computation overhead can be reduced by using symmetric key based cryptographic methods and hence significantly reduce the costs associated with the encryption and decryption operations. Elliptic curve based key agreement protocols significantly reduce the size of the keys in the agreement process and decreases the communication costs.

4.4 Key Agreement Interception

The security of the messages exchanged during the key agreement phase is very critical. The key agreement protocol is resistant towards passive interception of the messages because of the hardness of the Elliptic Curve Discrete Logarithm Problem [11]. The attacker Ian will have to perform a man-in-the-middle attack during the key agreement process. He can tamper with the messages between Alice and Bob, resulting in both having a modified unequal key. Subsequently, he can impersonate himself as either of Bob or Alice

and connect with the other device. Note that Alice and Bob will not be able to talk to each other in this case. Fig 4 shows how Alice and Bob end up with different keys that only Ian is aware of, in the Elliptic Curve Diffie-Hellman Key Agreement protocol. After the key exchange process, Alice and Bob will not be able to authenticate each other as they do not have the same key.

In our implementation, we used the Short Messaging Service (SMS) over the cellular network for the key exchange process. The Universal Mobile Telecommunication System (UMTS), an emerging standard for third generation (3G) wireless communications, has adopted an enhanced authentication and key agreement protocol [1]. The enhanced security of the 3G cellular systems enforces access control of users and mobile stations, data confidentiality, data integrity, and user identity privacy [5, 9, 10].

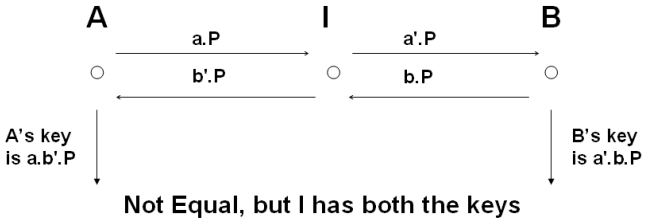

Figure 4. Man-In-The-Middle Attack during Key Agreement

Hence, in practice, it is sufficiently hard for an intruder to impersonate anybody else in the cellular network and tamper with messages. Ian will neither be able to impersonate himself as Alice or Bob, nor tamper with the messages exchanged during the key agreement protocol.

4.5 Loss of Device

In the event of the loss/theft of a device, the malicious user can connect to all the devices that are present in that device's address book. Though *SPACE* does not do anything explicitly to deal with this vulnerability, it assumes the existence of an external authority that deals with the general problems arising out of loss of a device and relies on the same for the security of our protocol. In practice, when a user loses his phone (and his SIM card), he notifies the phone manufacturer and the cellular provider who in turn have measures to track usage of the device and block the SIM card respectively.

5 Privacy Analysis

This section deals with the privacy implications of *SPACE*.

5.1 Hash Value based Identity

If the hash value broadcast in the Scan Phase was constant, an intruder can use it as the identity of a device to map all the places that the device was present e.g. to find if someone was in a particular location at different points in time; or if any one from a particular location moved to another location in the last ten minutes. So, if an intruder knows the hash of the contact detail of a device, he can easily locate all the places that the device visited by matching the hash value, thereby violating the privacy of the device even though he cannot invert the hash to obtain the contact detail. In our protocol, since we have a dynamic key for the hash function, the hash values are different for every broadcast.

5.2 Is any given contact detail present?

Assume that the adversary Ian wants to find if Bob is in Alice's address book. Ian can send a message pretending to be Bob. More generally, Ian can attempt to check the presence of a set of addresses in Alices address book.

In our protocol Alice responds with an encrypted nonce for the situation when she has Bob's address in her address book and does not send anything if the address is not present. This leaks the information Ian is seeking which causes a privacy concern. To address this we modify the protocol as follows. Now Alice sends a response in both situations:when she does not have the address she can encrypt with a random key. Ian will not be able to distinguish between the two responses. As before Alice will send an SMS if Bob's address was present but no key agreement was done. But if Ian is able to scan the atmosphere for SMS activity and analyze their pattern, he may be able to discover that Bob is indeed in the address book. We assume that Ian cannot scan the cellular network to isolate the needed cellular activity (SMS) from other normal activity when Alice exchanges messages corresponding to the key agreement protocol. Note that our extension costs an extra encryption and decryption step.

5.2.1 Are my contact details present?

A special case of the earlier attack is when Ian tries to determine the presence of his own address in Alices address book. An example is that authorities have a suspected mobile phone on hand and intend to find the set of people in a mall who have this number. In general, a clear policy has to be in place as to how to handle all the cases. Note that if Alice is in the mall, and Ian knows her number from the phone at hand, we expect Ian to detect her presence and make progress in setting up a connection.

Depending on whether Alice has Ian's address in her address book or not, Ian will receive the message corresponding to the key agreement protocol in the Authentication Phase. If this is a privacy concern for Alice, we can modify the Authentication phase in the protocol at an additional cost to Alice to send a challenge encrypted with her own address (this assumes address space is large enough, or is augmented to be large enough) before initiating the key agreement protocol. Alice initiates the key agreement protocol and the subsequent nonce challenge-response only if Ian responds correctly to the previously sent challenge. Otherwise she sends a nonce-challenge with a random key to Ian. Ian would not be able to respond to this challenge correctly and the protocol halts. But Ian will not be able to distinguish this random

challenge string from a valid challenge string and hence will not be able to make any conclusion about the presence of his address in Alice's address book.

If Alice did not have Ians contact detail he will not get any message corresponding to the key agreement protocol. But Ian cannot conclude from this because it could be that, either Alice does not have Ians contact detail or Alice already has an entry for Ian in his key table. So Ian cannot determine the presence of his contact detail in Alices address book.

6 IMPLEMENTATION

We implemented *SPACE* on smartphones running Windows Mobile 2003 SE operating system and communicate over Bluetooth (WLAN). The addresses we use are phone numbers in the cellular network. Alice hashes her phone number using SHA_1 and sends it to Bob. Bob verifies if the hash of any of the phone numbers in his phone book matches with the value sent by Alice. He subsequently sends over the hash of his number to Alice who performs the same check. In the first connection instance, Alice and Bob establish a shared secret key between them using the Elliptic Curve Diffie Hellman key agreement protocol [6] via SMS messages. The agreed key is stored for the purpose of authentication. The variables a , b and P used in the key agreement protocol were of sizes 160 bits each. Now, Alice and Bob can present a nonce to each other using this key and authenticate each other. We use the AES encryption algorithm for this purpose. The size of the nonces were 64 bits.

7 RELATED WORK

Bluetooth and IEEE 802.11x are the popular wireless local network standards. There has been considerable analysis on their security.

In Bluetooth, pairing is facilitated by the *initialization* key. This key is computed by a pair of devices using the Bluetooth addresses of each device, a random number, and a shared secret (PIN). The pairing session results in the *link* key that is unique for a pair of users and used for future communications. The security of the pairing process is dependent on the secrecy of the PIN. It is simple to crack the PIN if the communications occurring while the devices are paired is recorded [8]. Shaked et. al. [12] demonstrate that it is possible to crack a 4-digit PIN in 0.06 seconds using standard hardware. This makes it vulnerable to impersonation attacks. In contrast, SPACE does not suffer from these vulnerabilities and is specifically resistant to impersonation attacks.

In 802.11b authentication is performed by a challenge response procedure using a shared secret. After requesting authentication, the authenticator sends the initiator a 128-octet random number challenge. The initiator encrypts the challenge using the shared secret and transmits it back to the authenticator. A simple and powerful attack on this authentication mechanism is presented by Arbaugh et. al. [11]. First the intruder determines the pseudorandom string by recording the challenge (plaintext) and the response (ciphertext) and XOR-ing them. He then impersonates the victim by using the pseudorandom string to compute the response to subsequent challenges. This vulnerability towards plaintext/ciphertext attacks do not exist with SPACE as we do not send the unencrypted text at any stage.

8 CONCLUSIONS AND FUTURE WORK

We have proposed a novel protocol, *SPACE* for ad hoc connection establishment between mobile devices based on real-life relationships. We explored relevant security and privacy concerns with our protocol and augmented our protocol accordingly.

We understand that the address book entries are not the most secure basis for trust. There might be scenarios when we have contact details of people whom we do not entirely trust or not interested in sharing resources. We want to develop a mechanism to identify the preferred/trusted users in a contact list. One way is for users to manually mark the contact entries with whom they would prefer sharing resources. We can automate this by extracting information from the call-log/e-mails and obtain the set of contact entries with which a user has a high frequency of communication.

As a follow-up to Section 3 and 4, we aim to do a rigorous security and privacy analysis so that the protocol could be used for critical applications. We intend to incorporate a digital signature mechanism into the key agreement protocol and make it independent of the security policies of the underlying network. We also want to develop and test other useful applications on top of our protocol.

REFERENCES

[1] 3rd Generation Partnership Project; Technical Specification Group SA. Security Architecture, version 4.2.0, Release 4. In *3GPP*, 2001.

[2] A. A. Pirzada and C. McDonald. Establishing trust in pure ad-hoc networks. In *27th Australasian Computer Science Conference (ACSC) 26(1)*, pages 47–54, Jan 2004.

[3] Alfred J. Menezes , Scott A. Vanstone , Paul C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press Inc, Boca Raton, Florida, 1996.

[4] E. Thomas and G. Xydis. Security Comparison: Bluetooth Communications vs. 802.11.

[5] G. Koien. An introduction to access security in UMTS. In *IEEE Wireless Communication, vol. 11, no. 1, pp. 818*, Jun 2004.

[6] I. F. Blake, G. Seroussi and N. P. Smart. Elliptic curves in cryptography. Technical report, London Math. Soc. Lecture Note Series, 2000.

[7] Lauter K. The Advantages Of Elliptic Curve Cryptography for Wireless Security. In *IEEE Wireless Communications*, Feb 2004.

[8] M. Jakobsson and S. Wetzel. Security Weaknesses in Bluetooth. In *RSA Security Conference - Cryptographer's Track, LNCS 2020*, 2001.

[9] M. Shin et. al. Wireless Network Security and Interworking. In *IEEE, vol. 94, pp 455-466*, Feb. 2006.

[10] Third Generation Partnership Project. 3GPP Technical Specifications, 3G security; Security Architecture (Release 6). 2003.

[11] W. Arbaugh et. al. Your 802.11 Wireless Network has No Clothes. In *IEEE Wireless Communications*, December 2002.

[12] Yaniv Shaked and Avishai Wool. Cracking the Bluetooth PIN. In *3rd international conference on Mobile systems, applications, and services (MOBISYS '05)*, 2005.

[13] Zhaoyu Liu, Anthony W. Joy, Robert A. Thompson. A Dynamic Trust Model for Mobile Ad Hoc Networks. In *10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'04)*, pages 80–85, May 2004.