Talking To Strangers: Authentication in Ad-Hoc Wireless Networks

Dirk Balfanz, D. K. Smetters, Paul Stewart and H. Chi Wong
Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
{balfanz,smetters,stewart,hcwong}@parc.xerox.com

Abstract

In this paper we address the problem of secure communication and authentication in ad-hoc wireless networks. This is a difficult problem, as it involves bootstrapping trust between strangers. We present a user-friendly solution, which provides secure authentication using almost any established public-key-based key exchange protocol, as well as inexpensive hash-based alternatives. In our approach, devices exchange a limited amount of public information over a privileged side channel, which will then allow them to complete an authenticated key exchange protocol over the wireless link. Our solution does not require a public key infrastructure, is secure against passive attacks on the privileged side channel and all attacks on the wireless link, and directly captures users' intuitions that they want to talk to a particular *previously unknown device in their physical proximity. We have implemented our system in Java for a variety of different devices, communication media, and key exchange protocols.*

1. Introduction

Imagine the following situation: you are standing in an airport lounge and would like to print a sensitive document you just received on your wireless email gizmo. You can choose among a substantial number of printers set up in the lounge by various dotcoms, some familiar, some not. What you would like to do is choose a particular printer, and then make sure that your email gizmo prints to *that* printer – that no other printer, and no other traveler waiting in the lounge, can learn the contents of your sensitive document.

In the good old days, you would take out your printer cable, connect your email gizmo to your chosen printer, and be done with it. However, you would really prefer to accomplish this task using the wireless capabilities of both your email gizmo and the printer.

What can you do?

First, you need a way to let your email gizmo know how to find your desired printer over the wireless network. Assuming each printer had a unique name, you could type the name of the printer you want to use into your gizmo, or you could go through some sort of discovery protocol, and pick the correctly-named printer out of the list of responders. Second, you want a guarantee that your email gizmo is actually talking to the intended printer, and that the communication is secured.

If that printer had a certificate issued by an authority you trust, your email gizmo could, in theory, perform a key exchange with the printer and establish an authenticated and secret channel to it. Note several problems with this approach: first, we have to assume that there is an immense public key infrastructure in place – *every* printer (and every other potential participant in any ad-hoc network) has to have a unique name, and a certificate issued by an authority you trust. This is impractical and prohibitively expensive. Second, even with such an infrastructure there has to be a reliable way for you to find the name of the printer you want. We could imagine having labels that show the name of each printer, but then we would have to assume that no one tampered with those labels. Third, this procedure is not very user-friendly. It requires you to type cryptic names like `printer12345.fancyprint.com` into your email gizmo, or select correctly from a long list of similar names before you can print securely.

Without such a universal naming infrastructure, you might choose to go ahead and wirelessly exchange keys with the most likely candidate from your list. You would then have to make sure you had actually chosen correctly by comparing the fingerprint of the resulting shared secret displayed on your device with one displayed by the printer. This essential step is annoying and very likely to be skipped.

In this paper, we propose a cheaper, more secure, and more user-friendly solution to this problem (and to the problem of authentication in local ad-hoc wireless networks in general, for which our printing scenario merely serves as one example). In our example (see Figure 1),
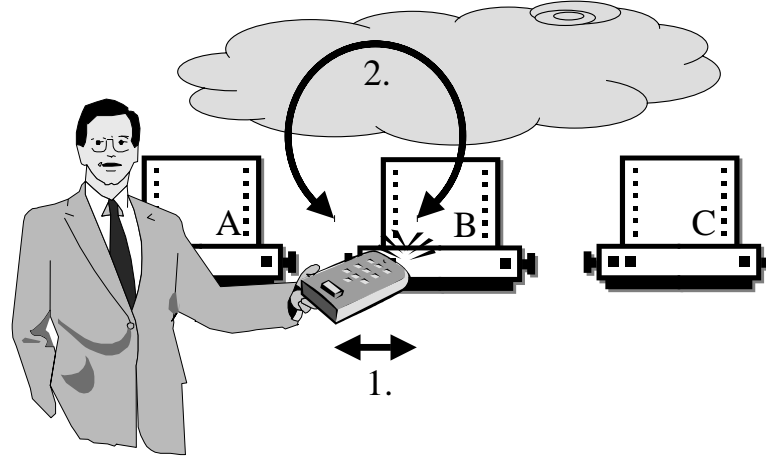
Figure 1. Pre-authentication and location-limited channels. The human operator introduces two devices, which (1) exchange pre-authentication information over a location-limited channel before they (2) authenticate each other over the wireless network.

you would walk up to the desired printer and briefly establish physical contact between it and your email gizmo. That will be enough for them to exchange their public keys. Your email gizmo can then proceed to perform a standard SSL/TLS key exchange with the printer over a wireless link (*e.g.,* Bluetooth or 802.11). Since it just learned to securely identify the printer's public key, it can verify that it is in fact talking to the right printer, and establish an authenticated and secret communication channel.

Such an exchange directly captures the user's intuition that s/he wants to communicate with *that* device by using a special, *location-limited* side channel to exchange a small amount of cryptographic information. That information can be used to authenticate standard key exchange protocols performed over the wireless link.

We would like to comment on a few concepts illustrated by this example:

Demonstrative identification. We identified the printer the email gizmo should talk to by the highly intuitive act of touching it. Contrast this with the clumsy way of identifying trusted communication parties on the Internet – in that case one usually has to type URLs into browsers. In the case of an ad-hoc wireless network where at least some of the participating devices are portable, you can simply walk up to a device and touch it. There is no need for a global public key infrastructure, certification authorities, or even names.

Location-limited channels. The printer and the email gizmo exchanged public information during physical contact. We call this physical contact a *location-limited channel*. Location-limited channels have the property that human operators can precisely control

which devices are communicating with each other. The notion of location-limited channels was introduced by Stajano and Anderson (although they did not use that name) [18], as a part of their "Resurrecting Duckling" model of interaction in ad-hoc networks. They use secret data exchanged over a contact channel to bootstrap a particular authentication and key exchange protocol ("imprinting" between a "mother" or control device, and a "duckling").

Pre-authentication. We can divide the "Duckling" protocol of Stajano and Anderson into two parts. In the first part, duckling and mother exchange secret information over a particular location-limited channel (physical contact). In the second phase, the duckling uses this secret data to recognize and authenticate its mother when she contacts it over the wireless link; the duckling is willing to be controlled by any "mother" that can present the right authentication data. We refer to the first phase as a *pre-authentication* exchange. The data that is exchanged over the location-limited channel during pre-authentication will then be used for subsequent authentication of the parties on the wireless link. We note that while Anderson and Stajano consider the use of such pre-authentication data as intrinsic to setting up a mother-duckling control relationship, that in fact it can be separated out and used in a wide variety of contexts.

In this paper, we generalize this idea of *pre-authentication* to secure arbitrary peer-to-peer ad-hoc interactions using a wide variety of key exchange protocols (as we saw in our example), and provide detailed blueprints for its use. We introduce the use of public key cryptography

in this context, and are thereby able to remove the secrecy requirement on location-limited channels used to authenticate key exchange protocols. This allows us to broaden the types of media suitable for use as location-limited channels to include, for example, audio and infrared. More importantly, it allows us to expand the range of key exchange protocols which can be authenticated in this manner to include almost any standard public-key-based protocol. As a result, our approach can be used with an enormous range of devices, protocols, and applications.

At the same time, our approach is significantly more secure than previous approaches, as we force an adversary to mount an active attack on the location-limited channel itself in order to successfully subvert an ad-hoc exchange. Previous approaches (*e.g.,* use of unauthenticated Diffie-Hellman key exchange) are either vulnerable to either active attacks in the main wireless channel, or, in the case of Anderson and Stajano, to passive (eavesdropping) attacks in the location-limited side channel [18].

The rest of this paper is structured as follows: In Section 2 we explain the notion of location-limited channels in more detail. We establish requirements for location-limited channels used for pre-authentication and discuss the security of the resulting composed authentication and key exchange protocols. In Section 3 we show concrete schemes for two-party pre-authentication. We first show how to use a pre-authentication stage to authenticate almost any well-established public-key-based key exchange protocol. We then present two additional schemes that may be of interest in situations where public key cryptography is considered an unacceptable computational burden. In Section 4 we explore the use of location-limited channels with broadcast characteristics (*e.g.,* audio) for pre-authentication in order to secure ad-hoc group communication. In Section 5 we briefly report on our first implementation before we wrap up with a comparison of related work in Section 6 and conclusions in Section 7.

2. Preliminaries

2.1. Location-Limited Channels and Pre-Authentication

Inspired by Anderson and Stajano [18], we propose bootstrapping secure wireless communication through pre-authentication over a location-limited channel. Location-limited channels are separate from the main wireless link, and have special security properties by virtue of the media over which data travels. In this section we examine what is required of such a channel, and list a number of existing technologies that can be used to implement one.

In order to be used for pre-authentication, a candidate location-limited channel must have two properties. First, it must support demonstrative identification; that is, identifi-

cation based on physical context (the printer in front of me, all the PDA's in this room, *etc.*). Communication technologies that have inherent physical limitations in their transmissions are good candidates. For example, audio (both in the audible and ultrasonic range), which has limited transmission range and broadcast characteristics, can be used by a group of PDAs in a room to demonstratively identify each other. For situations that require a single communication endpoint (*e.g.,* the printer across the room), channels with directionality such as infrared are natural candidates. It is these demonstrative properties that allow communication across a location-limited channel to "name" a target device or group of devices.

The second property required of a location-limited channel is *authenticity* – that it is impossible (or difficult) for an attacker to transmit in that channel, or at least to transmit without being detected by the legitimate participants. As we will see below, this property is sufficient to ensure that information exchanged over the location-limited channel will to allow the parties involved to securely authenticate each other over the wireless link, even in the presence of potential attackers.

A third property that was required in previous work is *secrecy* – that the channel be impervious (or resistant) to eavesdropping. For example, Anderson and Stajano [18] use *secret* data, such as a symmetric key, exchanged across the location-limited channel to allow participants to authenticate each other. As a result, that authentication protocol is vulnerable to a passive attacker capable of eavesdropping on the location-limited channel, thereby obtaining the secrets necessary to impersonate one of the legitimate participants. A location-limited channel used to exchange such secret pre-authentication data must therefore be very resistant to eavesdropping.[1]

If we can remove that requirement that pre-authentication data be secret, and instead only require that it be *authentic*, we can increase our security dramatically. Because legitimate participants would only communicate with entities from whom they had received pre-authentication data, we would now require an attacker to perform an *active* attack – to be able to transmit – not only in the main wireless medium, but also in the location-limited channel. Because of the physical limitations of transmission on location-limited channels, it is significantly harder for an attacker to passively eavesdrop on them, not to mention to actively transmit.

For such an active attack to succeed, the attacker must not only transmit on the location-limited channel, but must do so without being detected by any legitimate participant.

---

[1]Such a protocol may still be considerably more secure than one that does not use pre-authentication (*e.g.,* an unauthenticated key exchange over the wireless link), as the latter may be subject to active (or even passive, see [19]) attacks on the wireless link, which are considerably easier to mount.

To be effective, such detection does not require that we correctly identify the devices transmitting on the location-limited channel. Instead, it only requires one's ability to count: if you know that both you and your intended communication partner have successfully initiated communication (*e.g.,* the lights on the target device blink, the human using the other laptop indicates the communication attempt was successful), and you (or your proxy device) know that only two participants have attempted to inject messages into the location-limited channel, then you know you must be talking to whom you think you are. If something appears to be wrong, you can simply abort the communication protocol.

The difficulty of monitoring a pre-authentication for such unwanted participation depends on the type of channel used and the number of legitimate parties involved. The more directed the channel and the smaller the number of parties, the easier it is to monitor. Note that, because of the physical limitations of the channels used and this monitoring requirement, it is only possible to use our techniques to pre-authenticate devices that are physically co-located at the time of first introduction.

We therefore propose that any physically limited channel suitable for demonstrative identification, on which it is difficult to transmit without being detected by at least one legitimate participant (human or device), is a candidate for use as a pre-authentication channel. Such candidates include: contact, infrared, near-field signaling across the body (see [20]), and sound (both audible [16] and ultrasound). The amount of data exchanged across the pre-authentication channel is only a small fraction of that sent across the main wireless link, and so we can use channel media capable only of low data rates.

2.2. Use of Public Key Cryptography

How do we remove the requirement that pre-authentication data be kept secret? We can do this very simply through the use of public key cryptography. If the participants use the location-limited channel to exchange their public keys as pre-authentication data, it doesn't matter whether an attacker manages to eavesdrop on the exchange. The participants will authenticate each other over the wireless link by proving possession of their corresponding private keys; as the attacker does not know those private keys, he will not be able to impersonate any of the legitimate participants.

If we accept the existence of cryptographically-secure hash functions (*e.g.,* SHA-1), we can further limit the size of the pre-authentication data exchanged. The participants do not actually need to exchange their complete public keys as pre-authentication data, they merely need to commit to those keys (*e.g.,* by exchanging their digests).

2.3. Pre-Authentication of Established Key Exchange Protocols

Having described the use of location-limited channels to exchange pre-authentication data, we must now show how such data can be used to establish a secure and authenticated channel over the main wireless link. Instead of proposing novel protocols specific to this application, thereby introducing the security flaws endemic to new protocols, we prefer to provide general methods that allow the use of pre-authentication channels to bootstrap the use of *any* standard key exchange protocol to set up these secure and authenticated channels. This allows us to take advantage of all of the existing work in protocol design and security analysis. At the same time, we gain the advantages of our pre-authentication schemes in flexibility and ease of use, which are particularly important in the ad-hoc setting.

Combining pre-authentication with most standard public-key-based key exchange protocols is in fact, quite simple (see example in Figure 2). Almost all such protocols begin with the assumption that the participants already have access to authenticated copies of each other's public keys [14, 4, 6]. These protocols then provide methods to establish secure and authenticated channels, given that these public keys have already been exchanged – that the participants know *who* they are supposed to be talking to.

Pre-authentication schemes can be used to perform this initial step – to make sure that the legitimate participants get authenticated copies of each others' public keys. The participants exchange commitments to their public keys across a chosen location-limited channel. In doing so, they each identify *who* it is they wish to be communicating with – this is the purpose of "demonstrative identification". The exchange of pre-authentication data transforms this "demonstrative identification" step – *e.g.,* identifying the device you want to communicate with by touching it – into a form of identification that can be used to authenticate that device across the wireless link ("the device holding the private key corresponding to the public key committed to in this pre-authentication message").

The devices then contact each other on the wireless link, and exchange their complete public keys. This key exchange can either be prefixed to protocol execution, or (as in SSL/TLS) occurs naturally as a standard part of the chosen key exchange protocol. These keys are authenticated simply by virtue of the fact that they were the ones committed to across the pre-authentication channel. The devices now have authenticated copies of each others' public keys, which is what we need to proceed with our chosen established key exchange protocol on the wireless link. That protocol should ensure that the devices prove to each other that they indeed hold the private keys corresponding to their authenticated public keys.

If we assume that the data exchanged across the location-

Pre-authentication, taking place over the location-limited channel:

1. $A \rightarrow B:$ $addr_A, h(PK_A)$
2. $B \rightarrow A:$ $addr_B, h(PK_B)$

Authentication continues over the wireless channel with any standard key exchange protocol, *e.g.,* SSL/TLS:

1. $A \rightarrow B:$ TLS_CLIENT_HELLO

...and so on.

The various symbols denote:

$addr_A, addr_B:$	A's (resp. B's) address in wireless space, provided strictly for convenience
$PK_A, PK_B:$	the public key belonging to A (resp. B), either a long-lived key or an ephemeral key used only in this exchange
$h(PK_A):$	a commitment to PK_A, *e.g.,* a one-way hash of an encoding of the key

Figure 2. Basic scheme for pre-authentication.

limited channel is indeed authentic (that we would have detected any active attacks), and that the public key algorithm and the cryptographic hash function we chose are secure, then the security of the final composed protocol depends only on the security of the chosen key exchange protocol.

2.4. Security of Ad-Hoc Interactions

In choosing to engage in an ad-hoc network, you are effectively choosing to talk to strangers. As your mother may have warned you, there are some risks inherent in such a choice – and no cryptographic protocol, no matter how secure, can protect you from them. If you choose deliberately to communicate with a malicious adversary, that adversary can post your private messages onto a billboard somewhere. What we can do, and what we attempt to do in this paper, is ensure that when you choose to establish a connection to a previously unknown device, you are actually communicating, securely and authentically, with *that* device, and not an attacker in the next room.

3. Two-Party Protocols

In this section, we show concretely how to use pre-authentication to securely authenticate devices. First, we describe how to use this approach to authenticate almost any public-key-based key exchange protocol. This allows existing protocols (*e.g.,* SSL/TLS, IKE) to be used securely and easily in an ad-hoc setting. Second, because a significant number of devices in ad-hoc networks may not have the resources for doing public key operations, we also offer two cheaper alternatives. The first is a variant of our basic scheme, and requires only one of the parties to have public key. The second, which provides only integrity protection instead of secrecy, uses digests of pre-committed secrets to replace public keys.

3.1. Basic Protocol

In the most basic of our pre-authentication schemes, parties exchange commitment to their public keys over a location-limited channel. The information that is actually exchanged can be the public keys themselves, their certificates, or simply secure digests of the keys using cryptographic hash functions. The only requirement is that the information exchanged allows the receiver to verify the authenticity of the key that is used in the authentication protocol.

In Figure 2, parties exchange digests of their public keys in the pre-authentication phase. For convenience, each device can also transmit its address in wireless space (*e.g.,* a IP address and port number, or a Bluetooth device address) and a user-friendly name. We note, however, that the security of our scheme does not rely on the correctness of these additional data. (If you get a wrong IP address, for example, the party on the other end will not have the right private key, and will not be able to complete an authentication protocol with you.)

Once the pre-authentication is completed, the devices proceed to establish a secure connection between them over the main wireless link. To this end, they can use *any* established public-key-based key exchange protocol which requires them to prove possession of a particular private key (*e.g.,* SSL/TLS [4], SKEME [14] IKE [6], *etc.*), which in this case will correspond to the public key committed to in the pre-authentication step.

The choice of key exchange protocol may influence the exact form of the pre-authentication data exchanged, and in particular whether parties exchange their complete public keys or merely commitments to them. If the key exchange protocol used on the wireless link explicitly sends public keys or certificates, only commitments to those public keys need to be exchanged in pre-authentication. If instead it expects parties to already have each other's public keys,

Pre-authentication, taking place over the location limited channel:

1. $A \rightarrow B$: $addr_A, h(PK_A)$

2. $B \rightarrow A$: $addr_B, h(S_B)$

Authentication continues over the wireless channel, *e.g.*:

1. $A \rightarrow B$: PK_A

2. $B \rightarrow A$: $E_{PK_A}(S_B)$

...and so on.

Symbols as above, with the following additions:

S_B :   a secret belonging to B

$h(S_B)$:   a commitment to S_B, *e.g.*, a one-way hash of the secret

$E_{PK_A}(S_B)$:   the encryption of S_B under PK_A

Figure 3. Basic pre-authentication scheme modified to require only one public key.

then the keys themselves should be exchanged during pre-authentication. (If the location-limited channel does not have sufficient capacity, we can still send the commitments during pre-authentication, and prepend the keys themselves to the wireless exchange.)

Note that a party that does not receive pre-authentication data cannot authenticate its communication partner, and is therefore unprotected against impersonation. Thus, in most cases pre-authentication must be mutual – both parties must send and receive pre-authentication data (as in Figure 2).

There are some applications for which mutual authentication is not required. For instance, a device designed to provide a service to anyone that requests it does not need to authenticate its partner, and therefore would be the only one to send pre-authentication data. At the extreme, such a device could be a passive beacon (*e.g.*, an IR beacon or RFId tag), sending pre-authentication data sufficient to uniquely and securely identify its active proxy in wireless space. Such an approach could be used to add a measure of security and authentication to systems that use such beacons to provide a "digital presence" for physical objects [13].

Finally, this scheme is applicable to use either long-lived or ephemeral keys[2]. The choice is based entirely on the application at hand. In either case, the keys do not require certification by any trusted authority. If the key exchange protocol chosen requires the exchange of certificates, they can be self-signed.

3.2. Single Public Key Protocol

The basic scheme we proposed in Section 3.1 works if both devices are able to execute public key operations.

---

[2]Unlike long-lived keys, which are repetitively used across a number of key exchanges, ephemeral keys are made up afresh for each new transaction. They offer the advantage of anonymity because transactions using different keys cannot be linked together.

When only one of the devices has resources for expensive public key operations, we propose a less computationally expensive variant (Figure 3).

In Figure 3 only A has a public key PK_A; B has an arbitrary secret S_B instead (*e.g.*, a random number). As in the basic scheme, A sends a commitment to his public key during pre-authentication. As before, the commitment can be the public key itself, a certificate, or a digest of the key. B responds with a commitment to his secret, S_B, in the form of a digest $h(S_B)$ (as S_B is to remain secret, it cannot be sent in the clear and must be sent in digest form).

Once the pre-authentication is complete, they proceed with authentication. Party A sends its public key across the wireless channel. Party B verifies it against the commitment, and then uses it to encrypt S_B (and optionally other information used to construct a symmetric key) and returns the result to A. Such a protocol authenticates B by its ability to produce the secret S_B, and A by requiring it to prove its ability to decrypt that secret.

This scheme assumes that PK_A uses an algorithm for which encryption is computationally cheap (*e.g.*, RSA), so that the computational requirements on B are minimized. A protocol like SKEME [14] that authenticates participants by requiring them to prove their ability to decrypt a message would also be particularly amenable for use here.

3.3. Interactive Guy Fawkes Protocol

In cases where the devices involved are extremely limited in computational resources (public key operations are infeasible), and the available location-limited channels do not permit trusted exchange of secret data, we propose a new scheme for constructing a channel that provides authentication and integrity protection (though not encryption) of communication based entirely on cryptographic hash func-

Pre-authentication, taking place over the location-limited channel:

Round 0:
1. $A \rightarrow B$: $a_1 = h(\underline{A_1},h(X_2),X_1),\ h(X_1)$
2. $B \rightarrow A$: $b_1 = h(\overline{B_1},h(Y_2),Y_1),\ h(Y_1)$
3. $A \rightarrow B$: $h(b_1,X_1)$
4. $B \rightarrow A$: $h(a_1,Y_1)$

Authentication continues over the wireless channel:

Round 1:
1. $A \rightarrow B$: $\underline{A_1},\ h(X_2),\ X_1,\ a_2 = h(A_2,h(X_3),X_2)$
2. $B \rightarrow A$: $\overline{B_1},\ h(Y_2),\ Y_1,\ b_2 = h(\underline{B_2},h(Y_3),Y_2)$
3. $A \rightarrow B$: $h(b_2,X_2)$
4. $B \rightarrow A$: $h(a_2,Y_2)$

Round 2:
5. $A \rightarrow B$: $A_2,\ h(X_3),\ X_2,\ a_3 = h(A_3,h(X_4),X_3)$
6. $B \rightarrow A$: $\underline{B_2},\ h(Y_3),\ Y_2,\ b_3 = h(B_3,h(Y_4),Y_3)$
7. $A \rightarrow B$: $\overline{h(b_3},X_3)$
8. $B \rightarrow A$: $h(a_3,Y_3)$

Round 3:
9. $A \rightarrow B$: $A_3,\ h(X_4),\ X_3,\ a_4 = h(\underline{A_4},h(X_5),X_4)$
10. $B \rightarrow A$: $B_3,\ h(Y_4),\ Y_3,\ b_4 = h(\overline{B_4},h(Y_5),Y_4)$
11. $A \rightarrow B$: $h(b_4,X_4)$
12. $B \rightarrow A$: $h(a_4,Y_4)$

Round 4:
9. $A \rightarrow B$: $\underline{A_4},\ h(X_5),\ X_4,\ a_5 = h(A_5,h(X_6),X_5)$
10. $B \rightarrow A$: $\overline{B_4},\ h(Y_5),\ Y_4,\ b_5 = h(\underline{B_5},h(Y_6),Y_5)$
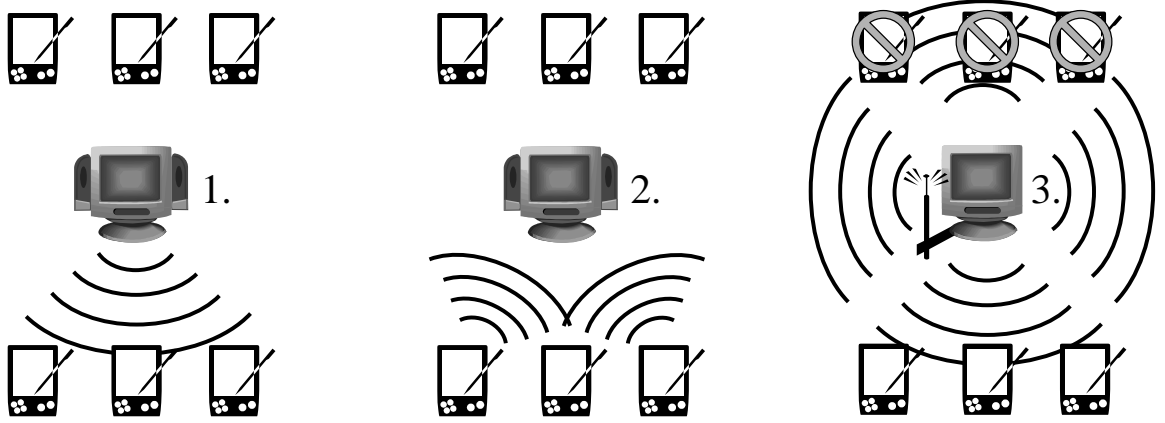11. $A \rightarrow B$: $h(b_5,X_5)$
12. $B \rightarrow A$: $h(a_5,Y_5)$

...and so on.

The various symbols denote:

X_i, Y_i : randomly generated data, used as authenticators
$h(Z_1,\ldots,Z_n)$: a one-way hash on the concatenation of values $Z_1,\ldots,Z_n$
A_i, B_i : Meaningless random message from A (resp. B) at round i
$\underline{A_i}, \underline{B_i}$: Meaninful message from A (resp. B) at round i
a_i, b_i : the commitment from A (resp. B) for round i

Figure 4. Interactive Guy Fawkes protocol

Figure 5. Pre-authentication over broadcast location-limited channels. (1) One device broadcasts pre-authentication information. (2) Human operators observe legitimate group members' response. If unwelcome devices respond, the protocol stops at this point. (3) After authentication and key exchange, each device may broadcast encrypted data, which can only be decrypted by legitimate group members.

tions.

Our proposal is based on the Guy Fawkes protocol [1], originally designed for authenticating digital streams. The Guy Fawkes protocol assumes that parties A and B want to exchange streams consisting of sequential blocks $A_0, A_1, A_2, \ldots$ and $B_0, B_1, B_2, \ldots$ respectively. At step i, A sends to B a packet P_i containing 4 pieces of data: block A_i; a random value X_i, used as an authenticator for A_i; the digest of the next authenticator $h(X_{i+1})$; and the digest of the message $a_{i+1} = h(A_{i+1}, h(Xi+2), X_{i+1})$. ($B$ does the same.) Assuming that B received an authenticated packet P_i, B can authenticate it as soon as it receives it, because P_i contained the digest $a_{i+1} = h(A_{i+1}, h(Xi+2), X_{i+1})$. Note that this claim does not hold if A and B do not execute in lock-step, and the authenticators are revealed before they should be (see [2] for details of such an attack). Finally, this protocol requires both A and B to know, one step ahead of time, what they want to say next, which makes the protocol unsuitable for interactive exchanges.

We modify the protocol to accommodate interactive communication. The key idea consists of having A (respectively B) commit to (and later send) a meaningless random message to B (respectively A) whenever A (respectively B) is not in a position to know what to say next. A is in such a position after he has sent a meaningful message, but before he has received a (meaningful) reply from B.

Figure 4 shows the modified protocol. Over the location-limited channel, A and B send the digest of the first secrets (authenticators) they will use to authenticate their first messages ($h(X_1)$ and $h(Y_1)$, respectively) together with the digests of their first messages (a_1 and b_1). They then continue the communication using the main wireless medium, re-

vealing the messages they committed to over the location-limited channel.

In round 1, as the initiator of the communication, A sends a meaningful message $\underline{A_1}$ to B. The reply B provides (B_1) is meaningless. It has to be meaningless because it was committed to in round 0, when B did not know message $\underline{A_1}$. In round 2, A sends a meaningless message A_2. It has to be meaningless because it is B's turn to "talk". B then sends $\underline{B_2}$, which is a meaningful message. Note that it was committed to in round 1, right after B learned the message $\underline{A_1}$. In round 3, A_3 is meaningless. It has to be because it was committed to in round 2, while A had not received $\underline{B_2}$. However, A can now commit to a meaningful message $\underline{A_4}$ because he has learned what B had to say in round 2. B_3 is meaningless because the next to "talk" is A. In round 4, A "talks" again and the protocol repeats itself. Note that the protocol does not actually require the presence of meaningless random messages to work: these messages can be replaced by empty messages.

Note that this interactive protocol, as well as the original non-interactive Guy Fawkes protocol, provides integrity protection and authentication, but cannot provide encryption. (See [1] for a security analysis of the Guy Fawkes protocol.) If the location-limited channel being used is believed to provide secrecy as well as integrity (*e.g.,* contact), it is possible to directly exchange a secret key across that channel, and use the key to encrypt further communications. However, such a direct exchange of secrets is vulnerable to passive eavesdropping in the location-limited channel, whereas the interactive Guy Fawkes protocol is not.

First, the key manager broadcasts its pre-authentication data over the location-limited channel:

1. $KM \xrightarrow{b} \text{group}$: $addr_{KM},\ h(PK_{KM})$

Then, group members send their pre-authentication data:

1. $A \rightarrow KM$: $addr_A,\ h(PK_A)$
2. $B \rightarrow KM$: $addr_B,\ h(PK_B)$
  . . .

The protocol continues over the wireless channel with any standard point-to-point key exchange protocol, *e.g.*:

1. $A \rightarrow KM$:   *TLS_CLIENT_HELLO*
2. $B \rightarrow KM$:   *TLS_CLIENT_HELLO*

...and so on; once connection is established the KM gives the appropriate multicast keys to every group member.

The various symbols denote:

$addr_A,\ addr_{KM}$: A's (resp. KM's) address in wireless space, provided strictly for convenience
$PK_A,\ PK_{KM}$:   the public key belonging to A (resp. B), either a long-lived key or an ephemeral key used only in this exchange
$h(PK_A)$:   a commitment to PK_A, *e.g.,* a one-way hash of an encoding of the key
$\xrightarrow{b}$:   message broadcast

Figure 6. Basic group key exchange protocol authenticated with local information.

4. Group Key Exchange Protocols

Some of the location-limited channels we have identified have broadcast capability – they can reach more than one target simultaneously. Using such broadcast channels, we can construct protocols that provide authenticated group communication. There are a number of applications that would benefit from the ability to rapidly and easily designate a group of users or devices to participate in a secure network – networked games and meeting support/conferencing software being the two most obvious.

Audio, in particular, is a medium that can provide significant advantages when used as a broadcast location-limited channel [16]. First, it can be monitored and tracked by humans – even if the people involved in the exchange do not know exactly what is carried in the audio messages, they can recognize that legitimate group participants ought to be sending them and the potted plant in the corner should not. Second, it can be incorporated into sounds that are already used by many pieces of software to provide feedback to humans – for instance, most corporate conference call settings play a short "join tone" whenever a new participant enters a call; such tones could be altered to also contain that participant's keying information. Third, because there are already designated channels designed to carry audio/voice information, it can actually be used via the telephone network (assuming one places reasonable trust in the carrier).

Audio is in some sense the canonical non-secret channel – it is literally possible to eavesdrop on communicated data. In using it as a pre-authentication channel we rely on the fact that our protocols are designed to be impervious to passive (eavesdropping) attackers. We defend against active attackers by emphasizing the ability of legitimate participants (human or device) to detect these illegitimate messages and abort the protocol (see Section 2).

As in the two-party case above, our goal is to use location-limited channels to authenticate secure key exchanges using well-established and trusted protocols. In this section, we investigate various options for pre-authenticating group communication. We will show how to use pre-authentication schemes with the two major families of group key exchange protocols: those that designate a specially-trusted group member, or "Group Manager", to distribute group keys, and those that do not.

4.1. Centrally Managed Groups

Figure 5 illustrates the setting for a centrally-managed group key exchange, and Figure 6 shows a straightforward example of a protocol involving pre-authentication. One participant is designated to become the manager (*e.g.,* the first to pre-authenticate, or a more complicated scheme can be used to elect a random participant). The group manager then establishes point-to-point links with every other participant using the two-party protocols described above. For efficiency, if the first member to broadcast is designated as the group manager, all participants after the first can use a digest-based authentication scheme (see Section 3.2).

In a centrally managed group, managing joining and leaving members is relatively easy. In the simplest possible approach, a joining member can use the two-party proto-

Each member broadcasts its pre-authentication data over the location-limited channel:

1. $A \xrightarrow{b}$ group : $\quad addr_A, h(PK_A)$

2. $B \xrightarrow{b}$ group : $\quad addr_B, h(PK_B)$

...

Participants exchange authenticated Diffie-Hellman public values over the wireless channel:

1. $A \xrightarrow{b}$ group : $\quad A, PK_A$

2. $B \xrightarrow{b}$ group : $\quad B, PK_B$

...

Participants continue with their chosen protocol to derive a shared secret key K:

1. $A \rightarrow B$: $\qquad PROTOCOL\_MSG\_1_{A,B}$
1. $B \rightarrow C$: $\qquad PROTOCOL\_MSG\_1_{B,C}$

...

The various symbols are as in Figure 6; the public keys PK_A, *etc.* are Diffie-Hellman public values.

Figure 7. Group key exchange protocol with no designated group manager.

cols discussed in Section 3.1 with the group manager to authenticate itself, and then receive the group key over a secured wireless channel. When a member leaves a group, the group manager can distribute a new group key to all remaining members over the wireless link. This is possible because the group manager has established shared secrets with all of the group members.

4.2. Unmanaged Groups

There are at least three problems with centrally managed groups. First, the group manager presents a single point of attack. Not only does it know the group key, it also knows shared secrets with every group member. Second, the group manager is trusted to generate and distribute all group keys; many applications are not compatible with such a distinguished trusted party. Third, the group manager cannot easily leave the group, since then there would be no one left to manage it. As a result, there is a large family of group key exchange protocols (*e.g.,* [12, 11]) designed to allow all members to equally participate in key generation, and hence all to be equally trusted. We would like to use our framework to authenticate this class of protocols as well.

Most group key-exchange protocols employ some sort of modified Diffie-Hellman key exchange among group members [12, 11]. However, just like two-party Diffie-Hellman, we know that we can establish a shared secret with *someone*, but we do not know necessarily who that someone is. As in the two-party case, these protocols assume that all group members participate in a shared public key infrastructure, or have previously exchanged public keys [12].

If we use pre-authentication over location-limited channels, these assumptions no longer have to be made. We can use a broadcast location-limited channel to allow all group participants to commit to their public keys publically to one or more group members. Group members can then proceed with their chosen group key exchange protocol over the wireless link using these authenticated keys. In Figure 7, we show an example of such an exchange.

Group members who join asynchronously can broadcast their key commitments over the location-limited channel to the rest of the group as they arrive, and a randomly selected current group member can respond, thus ensuring mutual authentication. The chosen group key exchange protocol is used to handle the details of updating the shared group key for these new group members, or revoking keys of departing members.

5. Implementation

We have begun experimenting with these protocols as part of a larger project investigating new paradigms for usable security. We have implemented the basic protocols in $Java^{TM}$, built to provide a flexible substrate for exploring many of the pre-authentication methods discussed in this paper.

We have built a software framework for using pre-authentication data to authenticate arbitrary key exchange protocols. This framework allows dynamic choice of the medium used for the location-limited channel, the public key algorithm used for the key commitments, and the final authenticated key exchange protocol used over the wireless link (and in fact, this stage assumes only a TCP/IP socket, allowing pre-authentication data to be used to authenticate secure connections made over a wired network as well). Extending the framework to provide a new location-limited channel type, or a new key exchange protocol,

is only a matter of implementing a $Java^{TM}$ interface to provide a small amount of syntactic "glue". The framework provides both client and server components, and allows developers to choose from either low-level, step-by-step control over data exchange, or to use simpler, higher-level interfaces. Such interfaces, for instance, provide server threads that can manage pre-authentication of multiple clients over the location-limited channel, and offer control over how such pre-authentication data is used to authenticate those clients over the wireless link (*e.g.,* serially, where only the most recent client to pre-authenticate is allowed to connect wirelessly, or in a multicast configuration where all pre-authenticated clients are allowed to connect at once). Framework components maintain state tracking who has currently pre-authenticated, what keying information is currently in use by this endpoint, *etc.*

An example scenario implemented using this framework consists of a client (such as our email gizmo from the example above), which is the initiator of the authenticated channel, and a responding server. The server component listens for a connection on both the location-limited channel and the primary link, but only admits primary-link connections from clients who have performed pre-authentication on the location-limited channel.

We currently use IrDA [9] as the medium for the location-limited channel. We are in the process of constructing a contact-mediated interface, and plan to expand shortly to group authentication using audio. The client opens an IrDA connection to the server (generating an error if it discovers more than one potential IrDA endpoint). Across this connection client and server exchange XML-encoded pre-authentication data containing a commitment to an ephemeral DSA public key, a "friendly name", and an IP address and a port on which the server will be listening. This yields a payload of the order of 300 bytes in each direction. With such a small payload size, the pre-authentication step incurs very little time overhead even on low-bandwidth location-limited channels.

With the pre-authentication complete, the IR channel is closed, and the client extracts the server's IP address and port number from the data it received. The client then opens a normal SSL/TLS connection to the server on the primary link. Each side uses the information gained in the pre-authentication step (namely the commitments to the public keys) to authenticate the newly opened channel. The client and server are now free to securely exchange any information they choose over the primary (wireless) link.

6. Related Work

Our work addresses the problem of bootstrapping trust in networked environments. Traditional solutions to this problem (*e.g.,* X.509 [7]) link a target to some cryptographic information (*e.g.,* a key pair) through some out-of-band mechanism, and then use that cryptographic information to securely identify the target. Trusted web server certificates that link domain names to key pairs are an example of such a mechanism. Approaches that rely on certificates for bootstrapping trust require heavy setup and online servers, and are therefore inappropriate for wireless ad-hoc networks.

A number of approaches to trust and key management have used out-of-band channels to authenticate key exchanges. In the simplest version of PGP's web of trust [21], users obtain public keys from a variety of insecure sources (*e.g.,* web sites, key servers, *etc.*). To make sure the key that they receive is authentic, users then engage in some out-of-band communication (*e.g.,* phone, US mail, face-to-face conversation or exchange of business cards) with the party they believe to be the key's owner to obtain the *fingerprint*, or digest, of the key, which they can then use to judge the authenticity of the key they obtained insecurely. They trust the fingerprint because it was obtained over a secure channel. The act of getting the fingerprint of a public key over the phone in PGP is, in essence, a pre-authentication step. What distinguishes PGP's pre-authentication approach from ours are: the types of out-of-band channels used; the type of entity (device or human) verifying the pre-authentication data; and whether the verification of pre-authentication data is a separate, optional (and frequently skipped) step in key exchange, or is built in as a seamless part of the key exchange itself. In our approach we use the exchange of pre-authentication data in part for *demonstrative identification*, to select our desired communication partner at the same time as we automatically authenticate them. PGP attempts instead to link keys to email addresses (names), and adds on manual key authentication (fingerprint comparison) as a separate step.

A few proposals have been put forth recently to address the issue of bootstrapping trust in the specific context of ad-hoc wireless networks. Bluetooth [3], in its most secure configuration, requires the user to enter a (preferably long and random) PIN into both devices to bootstrap their first communication. This PIN serves as the out-of-band information, but puts a burden on users. Aside from usability issues, Bluetooth is plagued by a wide variety of other security flaws [10]. WEP, the link-layer security protocol for 802.11 [8], has the same usability issues. It requires a group of communicating devices to be initialized with the same key, usually derived from a password. WEP too has been broken [5, 19]. Our proposal is more appealing than Bluetooth and WEP from both usability and security points of view. Our pre-authentication step is intuitive and user-friendly, and we rely on well-known and tested protocols for key exchange.

Stajano and Anderson [18] suggested the use of an out-of-band mechanism for establishing trust when they pro-

posed the Resurrecting Duckling security model to regulate secure transient association between devices in ad-hoc wireless networks where authentication servers may not be available. In their model, a master-slave relationship between two devices is set up when the master (or the mother, in their terminology) establishes a shared secret with the slave (the duckling) through a contact channel. This shared secret will enable the duckling to recognize the mother and be controlled by her in future interactions. Stajano later extended this model to address peer-to-peer interactions [17]. In the extended model, the mother can upload an access-control policy into the duckling. This policy then determines the type of relationships that the slave can have with other devices (other than the mother). Our work extends theirs in a number of ways: while Anderson and Stajano suggest the use of contact channels to exchange secret authentication data, they don't provide any details of what that data should be or how to combine it appropriately with data sent on the wireless link. We have provided those concrete details here. We separate the very general idea of pre-authentication for bootstrapping security in ad-hoc networks from their very specific notion of mother-duckling "imprinting", and show that it can in fact be used to secure a wide variety of protocols and applications. We extend their work through the use of public key cryptography, allowing us to take advantage of a much wider range of privileged channel types, and to make use of well-established key exchange protocols. And finally, for those situations where the computational load of public key cryptography is unacceptable, we provide cheaper, hybrid options that share many of the advantages of the public key schemes.

Outside the security domain, location-limited channels have been used as a means for accessing devices and services demonstratively. Satchel/MobileDoc [15] from XRCE allows users to use a PDA to retrieve documents located in their home offices, and "beam" them to a printer, a PC, or another wireless PDA. In HP's Cooltown project [13], entities in a user's surroundings have web presences, as well as physical tags that send out their corresponding URL's through infrared. To interact with such an entity, the user first point her wireless device to the entity's tag, receiving its URL, and then proceed with a (wired or wireless) interaction with the entity using the URL. There have even been attempts to standardize these approaches. The IrDA [9] has been creating and promoting interoperable, infrared connection standards that support a walk-up, point-to-point user model. These efforts all recognize the usability advantages of demonstrative identification, but make no provision for security. Our proposal provides a way to simply and seamlessly add security to these efforts without increasing the demand on the user.

7. Conclusions

In this paper we presented new schemes for peer-to-peer authentication in ad-hoc wireless networks. Building on previous work by Stajano, Anderson, and others, we explained how to use demonstrative identification to perform pre-authentication over location-limited channels. Demonstrative identification provides the user with an extremely intuitive way to identify – and authenticate – parties to a communication. Our schemes do not require a public key infrastructure, and do away with the naming problem that plagues traditional authentication systems. Below we summarize the novel aspects of our work:

Use of location-limited channels. We propose the use of location-limited channels to bootstrap a wide range key-exchange protocols. In particular, we do not limit ourselves to imprinting a duckling device with its mother's secret key.

Novel location-limited channels. Because our location-limited channels do not have to provide secrecy, we open the door to new media. In our prototype we are currently experimenting with audio, infrared, and contact-based channels, but other media are certainly imaginable.

Concrete pre-authentication protocols. We provide a concrete recipe for augmenting existing key exchange protocols with a pre-authentication step. For the case that both communicating parties are incapable of public key operations, we also introduce a new, interactive, version of the Guy Fawkes protocol. We note that none of the protocols presented exchange any secret information. Therefore, a passive attacker cannot gain anything by eavesdropping on the location-limited channel. We raise the security bar by requiring the attacker to become active in the location-limited channel, and we explained how active attacks can be detected by human operators or by the system.

Group communication. Because we use location-limited channels that are not necessarily secret, we can employ broadcast characteristics of some media (such as audio) to pre-authenticate group communication over location-limited channels.

No reliance on Public Key Infrastructure. Key exchange and key agreement protocols depend on an authentication step to verify *who* we are exchanging a key with. Public Key Infrastructures (PKI) have been commonly suggested as a way of solving this authentication problem. PKIs use a trusted authority to bind public keys to names or other identifiers; those names are used in turn to identify the party with whom you wish to communicate (*e.g.,* SSL/TLS implementations

require X.509 certificates that certify a web server's DNS *name* and are signed by a certification authority trusted by the web client). We have shown that such a reliance on pre-existing third party naming and trust infrastructures is unnecessary if one can briefly bring communicating parties within close physical proximity. In such a case, our pre-authentication protocols can be used in place of a PKI.

References

[1] Anderson, Bergadano, Crispo, Lee, Manifavas, and Needham. A new family of authentication protocols. *ACMOSR: ACM Operating Systems Review*, 32, 1998.

[2] S. M. Bellovin and M. Merrit. An attack on the interlock protocol when used for authentication. *ACM Transactions on Information Theory*, 40(1), January 1994.

[3] The official bluetooth SIG website. `www.bluetooth.com`.

[4] T. Dierks and C. Allen. *The TLS Protocol Version 1.0*. IETF - Network Working Group, The Internet Society, January 1999. RFC 2246.

[5] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of RC4. In *Eight Annual Workshop on Selected Areas in Cryptography*, August 2001.

[6] D. Harkins and D. Carrel. *The Internet Key Exchange (IKE)*. IETF - Network Working Group, The Internet Society, November 1998. RFC 2409.

[7] R. Housley, W. Ford, W. Polk, and D. Solo. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. IETF - Network Working Group, The Internet Society, January 1999. RFC 2459.

[8] IEEE. ANSI/IEEE. 802.11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1999.

[9] IrDA Association. Technical summary of IrDA DATA and IrDA CONTROL, 1999.

[10] M. Jakobsson and S. Wetzel. Security weaknesses in bluetooth. In *Topics in Cryptology - CT-RSA 2001*, volume 2020, pages 176–191, San Francisco, April 2001. Springer.

[11] M. Just and S. Vaudenay. Authenticated multi-party key agreement. In *Advances in Cryptology – ASIACRYPT '96*, Lecture Notes in Computer Science, Berlin, 1996. Springer-Verlag.

[12] Y. Kim, A. Perrig, and G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. In S. Jajodin and P. Samarati, editors, *7th ACM Conference on Computer and Communications Security*, pages 235–241, 2000.

[13] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra, and M. Spasojevic. Places and things: Web presence for the real world. Technical Report HPL-2000-16, HP Labs, 2000.

[14] H. Krawczyk. SKEME: A versatile secure key exchange mechanism for internet. In *Proceedings of the 1996 Network and Distributed Systems Security Symposium (NDSS'96)*, pages 172–194, San Diego, CA, February 1996. The Internet Society.

[15] M. Lamming, M. Eldridge, M. Flynn, C. Jones, and D. Pendlebury. Satchel: Providing access to any document, any time, anywhere. *ACM Transactions on Computer-Human Interaction*, 7(3):322–352, 2000.

[16] C. Lopes and P. Aguiar. Aerial acoustic communications. In *Proceedings of the 2001 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, October 2001.

[17] F. Stajano. The resurrecting duckling - what next? In *Security Protocols—8th International Workshop*, Lecture Notes in Computer Science, Cambridge, United Kingdom, Apr. 2001. Springer-Verlag, Berlin Germany.

[18] F. Stajano and R. J. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *7th Security Protocols Workshop*, volume 1796 of *Lecture Notes in Computer Science*, pages 172–194, Cambridge, United Kingdom, 1999. Springer-Verlag, Berlin Germany.

[19] A. Stubblefield, J. Ioannidis, and A. D. Rubin. Using the Fluhrer, Mantin, and Shamir Attack to Break WEP. In *Proceedings of the 2002 Network and Distributed Systems Security Symposium (NDSS'02)*, San Diego, CA, February 2002. The Internet Society.

[20] T. G. Zimmerman. Personal Area Networks: Near-field intrabody communication. *IBM Systems Journal*, 35(3&4):609–617, 1996.

[21] P. R. Zimmermann. *The Official PGP User's Guide*. MIT Press, Cambridge, MA, USA, 1995. ISBN 0-262-74017-6.