Distance-Bounding Protocols

(Extended abstract)

Stefan Brands[1] and David Chaum[2]

[1] CWI, Amsterdam, email: brands@cwi.nl
[2] CWI & DigiCash, Amsterdam, email: david@digicash.nl

Abstract. It is often the case in applications of cryptographic protocols that one party would like to determine a practical upper-bound on the physical distance to the other party. For instance, when a person conducts a cryptographic identification protocol at an entrance to a building, the access control computer in the building would like to be ensured that the person giving the responses is no more than a few meters away.
The "distance bounding" technique we introduce solves this problem by timing the delay between sending out a challenge bit and receiving back the corresponding response bit. It can be integrated into common identification protocols. The technique can also be applied in the three-party setting of "wallets with observers" in such a way that the intermediary party can prevent the other two from exchanging information, or even developing common coinflips.

1 Introduction

A prover convincing a verifier of some assertion is a frequently recurring element in many applications of cryptography. One potentially useful such assertion is that the prover is within a certain distance. It seems that this problem has not been specifically adressed, let alone solved in the literature. We introduce a technique called "distance bounding" that enables the verifying party to determine a practical upper-bound on the physical distance to a proving party.

In the literature, so-called "mafia frauds" have been adressed in which a party identifies himself to a verifying party using the identity of a third party, without that third party being aware of it. With our distance-bounding technique we can prevent these frauds as a special case.

Our distance-bounding protocols can be integrated with known public-key identification schemes, such that the verifier cannot obtain information that he could not have computed himself.

In the recently proposed setting of "wallets with observers," distance bounding can be incorporated in such a way that the verifying party can determine a practical upper-bound to the observer, whereas the intermediary party can prevent the other two parties from exchanging or developing information which can be used to compromise privacy.

This paper is organized as follows: In Section 2 we introduce the distance-bounding principle. We introduce our solution in parts and then unify them.

T. Helleseth (Ed.): Advances in Cryptology - EUROCRYPT '93, LNCS 765, pp. 344-359, 1994.

In Section 3, we describe how distance bounding can be integrated into known public key identification schemes. In Section 4, we describe a problem in the setting of wallets with observers. We then show how to use the distance-bounding technique to solve it. A final section ends this paper with some open problems.

2 Distance-bounding protocols

In this section, we first present the basic distance-bounding principle. We then discuss mafia frauds and previously proposed countermeasures. We show how distance bounding can be used to prevent these frauds. We go on to show how distance bounding can prevent frauds in which a party having access to the secret keys convinces a verifying party that he is within a certain distance whereas he is not. Both protocols are then merged into one protocol that prevents both attacks.

2.1 The distance-bounding principle

The essential element of a distance-bounding protocol is quite simple. It consists of a single-bit challenge and rapid single-bit response. In practice, a series of these rapid bit exchanges is used, the number being indicated by a security parameter k. Each bit of the prover $\mathcal{P}$ is to be sent out immediately after receiving a bit from the verifier $\mathcal{V}$. The delay time for responses enables $\mathcal{V}$ to compute an upper-bound on the distance.

What makes this approach really practical is that today's electronics can easily handle timings of a few nanoseconds, and light can only travel about 30cm during one nanosecond. For instance, even the timing between two consecutive periods of a 50 Mghz clock allows light to travel only three meters and back. (Later on we introduce exclusive-or operations on the bits exchanged, but 10113 chips have several such gates each with a throughput of two nanoseconds.)

2.2 Mafia frauds

A *mafia fraud*, first described in [9], is a real-time fraud that can be applied in zero-knowledge or minimum disclosure identification schemes by fraudulent prover $\overline{\mathcal{P}}$ and verifier $\overline{\mathcal{V}}$, cooperating together. It enables $\overline{\mathcal{P}}$ to convince an honest verifier $\mathcal{V}$ of a statement related to the secret information of an honest prover $\mathcal{P}$, without actually needing to know anything about this secret information. To this end, when $\mathcal{P}$ is about to perform the protocol with $\overline{\mathcal{V}}$, the latter establishes, say, a radio link with $\overline{\mathcal{P}}$, and will send any information transmitted to him by $\mathcal{P}$ straight on to $\overline{\mathcal{P}}$, who in turn sends it on to $\mathcal{V}$. The same strategy is applied by $\overline{\mathcal{P}}$, who sends information received from $\mathcal{V}$ on to $\overline{\mathcal{V}}$, who in turn sends it on to $\mathcal{P}$. In effect, $\overline{\mathcal{V}}$ and $\overline{\mathcal{P}}$ act as a single transparent entity, sitting in the middle between $\mathcal{P}$ and $\mathcal{V}$. This enables $\overline{\mathcal{P}}$ to identify himself to $\mathcal{V}$ as being $\mathcal{P}$, without any of $\mathcal{P}$ and $\mathcal{V}$ noticing the fraud.

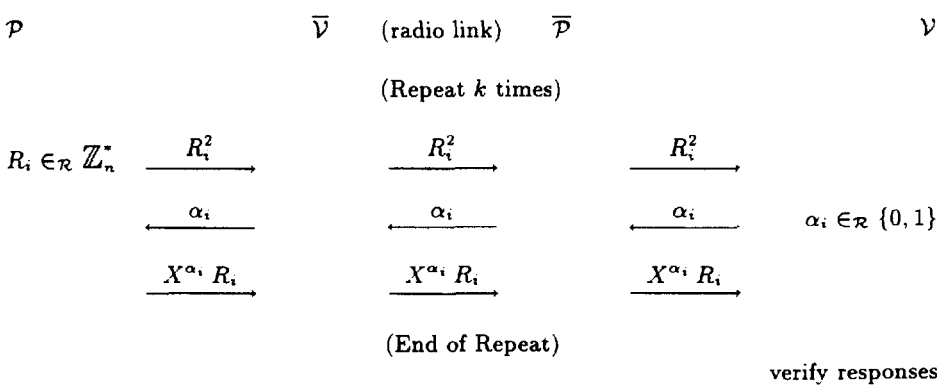

Fig. 1. A mafia fraud in the basic Fiat-Shamir identification scheme.

In Figure 1, a mafia fraud is shown as it would be applied in the basic Fiat-Shamir identification scheme (see [12]). In order to enhance readability of the figures, we define the subscript i to run over the set $\{1, \ldots, k\}$, and computations are modulo n. In the most basic form of the Fiat-Shamir scheme $\mathcal{P}$ identifies himself to $\mathcal{V}$ by proving knowledge of a square root X of $X^2 \bmod n$, where $X^2 \bmod n$ in some way is related to $\mathcal{P}$'s identity or has been published in a trusted directory. As usual, n is the product of two distinct primes.

In [9], Desmedt proposed a countermeasure to mafia frauds which requires $\mathcal{P}$ to sign a message that contains his physical location on earth, and then prove to $\mathcal{V}$ knowledge of the signature. Usually in an identification scheme, $\mathcal{P}$ will be represented by some user-module, so it will be impracticable to implement this solution without requiring position detection or cooperation of the user. Also, it cannot guarantee that the verifier in the long run does not learn anything about the secret key of the prover.

In [2], Beth and Desmedt propose that all transmission times be accurately measured. This seems to be useless owing to the significant possible variations in speed of computation.

In [1], Bengio *et al* suggested that $\mathcal{V}$ shield $\mathcal{P}$'s module from the outside world (e.g., in a Faraday cage) when the protocol takes place. This countermeasure requires trust by $\mathcal{P}$ that $\mathcal{V}$ does not secretly modify his module in some way while shielded. One would rather like to identify oneself in such a way that the module remains visible (an infrared channel would be even better, the user-module never needing to leave the hands of the user). Futhermore, it requires special hardware equipment.

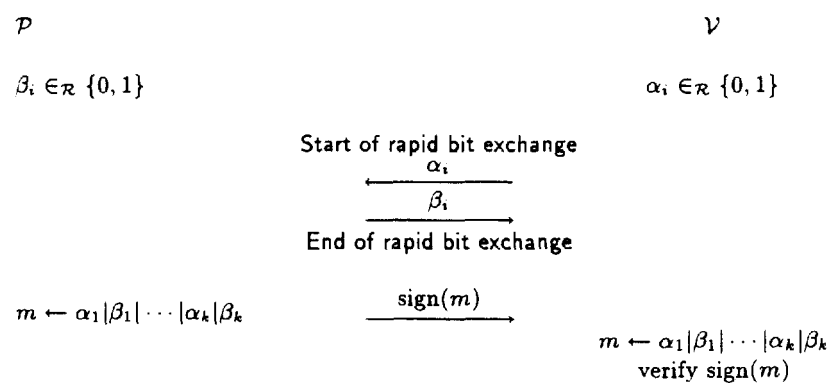

Fig. 2. Distance bounding to prevent mafia frauds.

2.3 Preventing mafia frauds using distance bounding

Consider how the distance-bounding principle can be used to prevent mafia frauds. We can assume that the distance between $\mathcal{P}$ and the fraudulent parties is not less than the accuracy that can be achieved with the apparatus being used, since otherwise obvious countermeasures can be taken. To ensure that the distance between $\mathcal{V}$ and the party $\mathcal{P}$ having access to the secret keys is measured, after the rapid bit exchanges have taken place the message formed by concatenating all the $2k$ bits sent back and forth in the distance-bounding stage is signed by $\mathcal{P}$, using his secret key (see Figure 2):

Step 1 $\mathcal{V}$ generates uniformly at random k bits α_i, and $\mathcal{P}$ generates uniformly at random k bits β_i. (Note: this can take place well beforehand.)

Step 2 Now the low-level distance-bounding exchanges can take place. The following two steps are repeated k times, for $i = 1, \ldots, k$.

- $\mathcal{V}$ sends bit α_i to $\mathcal{P}$.
- $\mathcal{P}$ sends bit β_i to $\mathcal{V}$ *immediately after* he receives α_i.

Step 3 $\mathcal{P}$ concatenates the $2k$ bits α_i and β_i, signs the resulting message m with his secret key, and sends the signature to $\mathcal{V}$. We denote concatenation by the symbol "|."

Now $\mathcal{V}$ can determine an upper-bound on the distance to $\mathcal{P}$ using the maximum of the delay times between sending out bit α_i and receiving bit β_i back, for $i = 1, \ldots, k$. $\mathcal{V}$ accepts if and only if $\mathcal{P}$ is close by, and the received signature is a correct signature of $\mathcal{P}$ on $m = \alpha_1|\beta_1|\cdots|\alpha_k|\beta_k$.

Proposition 1. *If the signature scheme is secure and $\mathcal{P}$ is not close by to $\mathcal{V}$, then a mafia fraud has probability of success at most $1/2^k$.*

That is, the probability of successful cheating decreases exponentially in the number of repetitions of the rapid bit exchange. The simple proof of this proposition is very similar to the proof of Proposition 3 in the next section.

2.4 Preventing the prover from sending bits out too soon

In this subsection we study a setting in which $\mathcal{P}$ has access to the secret keys, and $\mathcal{V}$ wants to be ensured that $\mathcal{P}$ is close by. A remarkable thing about the distance bounding stage in the protocol of the previous subsection is that the bits that $\mathcal{P}$ sends to $\mathcal{V}$ do not have to depend on the bits that $\mathcal{V}$ sends to $\mathcal{P}$. If $\mathcal{P}$ knows at what times $\mathcal{V}$ will send out bits, he can have $\mathcal{V}$ accept by sending β_i out to $\mathcal{V}$ at the correct time before he receives α_i, regardless of the distance to $\mathcal{V}$. Hence, the protocol we described for preventing mafia frauds does not prevent this fraud.

Two solutions suggest themselves. The first solution consists of $\mathcal{V}$ sending bits out with randomly chosen delay times. Since $\mathcal{P}$ cannot anticipate when $\mathcal{V}$ expects to have received back a bit, he cannot send out bits β_i before he has received bit α_i (since $\mathcal{V}$ will not accept if a response bit β_i arrives before he has sent out bit α_i). In fact, it is sufficient if $\mathcal{V}$ sends out bit α_i at random at one of two discrete times, say, at the rising edge of clock pulse $3i$ or $3i+1$, for $1 \leq i \leq k$. The probability of the strategy having success is at most $1/2^k$ if the choices are made independently.

The second solution consists of ensuring $\mathcal{V}$ that $\mathcal{P}$ must choose bits β_i depending on α_i. One way to do this involves creating a public bitstring $m_1|\cdots|m_k$ once (the choice of the bits m_i is irrelevant). The following protocol implements this (see Figure 3):

Step 1 $\mathcal{V}$ generates uniformly at random k bits α_i.

Step 2 Now the low-level distance-bounding exchanges can take place. The following steps are repeated k times, for $i = 1, \ldots, k$.

- $\mathcal{V}$ sends bit α_i to $\mathcal{P}$.
- $\mathcal{P}$ sends bit $\beta_i = \alpha_i \oplus m_i$ to $\mathcal{V}$ *immediately* after receiving bit α_i from $\mathcal{V}$.

$\mathcal{V}$ verifies whether the bit-string $(\alpha_1 \oplus \beta_1)|\cdots|(\alpha_k \oplus \beta_k)$ equals the public bitstring. If so, $\mathcal{V}$ computes an upper-bound on the distance to $\mathcal{P}$ using the maximum of the delay times between sending out bit α_i and receiving bit β_i back, for $i = 1, \ldots, k$. $\mathcal{V}$ accepts if and only if $\mathcal{P}$ is close by.

As before, it is easy to see that the probability that $\mathcal{V}$ accepts when $\mathcal{P}$ is not close by is at most $1/2^k$.

2.5 Preventing both types of fraud

By combining the two protocols, we can prevent both types of fraud. As before, it is assumed that a bit-string $m_1|\cdots|m_k$ is published. The following protocol can be used (see Figure 4):

Step 1 $\mathcal{V}$ generates uniformly at random k bits α_i.

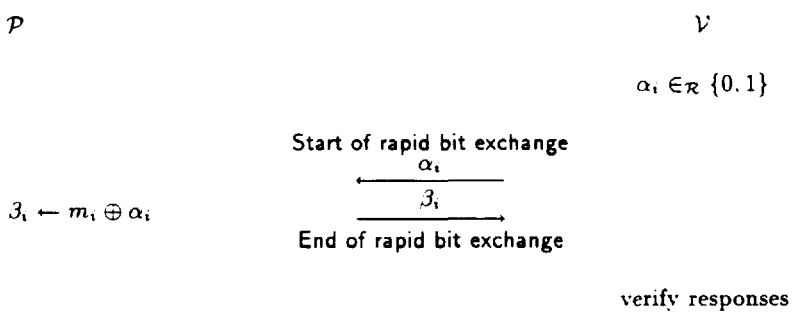

Fig. 3. Preventing the response bits from being sent out too soon.

Step 2 $\mathcal{P}$ generates uniformly at random k bits m_i. As before, both $\mathcal{P}$ and $\mathcal{V}$ can do so well beforehand. $\mathcal{P}$ commits to k bits m_i using a secure commitment scheme.

Step 3 Now the low-level distance-bounding exchanges can take place. The following steps are repeated k times, for $i = 1, \ldots, k$.

- $\mathcal{V}$ sends bit α_i to $\mathcal{P}$.
- $\mathcal{P}$ sends bit $\beta_i = \alpha_i \oplus m_i$ to $\mathcal{V}$ *immediately* after he receives α_i.

Step 4 $\mathcal{P}$ opens the commitment(s) on the bits β_i by sending the appropriate information to $\mathcal{V}$. $\mathcal{P}$ concatenates the $2k$ bits α_i and β_i, signs the resulting message m with his secret key and sends the resulting signature to $\mathcal{V}$.

With the information received in Step 4, $\mathcal{V}$ verifies whether the bits $\alpha_i \oplus \beta_i$ are indeed those commited to in Step 2. If this holds, then $\mathcal{V}$ computes m in the same way as $\mathcal{P}$ did and verifies whether the signature he received is indeed a correct signature of $\mathcal{P}$ on m. If so, he computes an upper-bound on the distance to $\mathcal{P}$ using the maximum of the delay times, and accepts if and only if $\mathcal{P}$ is close by.

3 Integration with public key identification schemes

The fact that a secure signature scheme must be used in the protocols of Subsection 2.3 and 2.5 can be a problem when the prover wishes only to identify himself by for example proving knowledge of a square root X of X^2 mod n (as in the basic Fiat-Shamir identification scheme): it is not clear how he should sign the message by using his secret information X; also, since $\mathcal{V}$ receives information that he could not have computed himself, it is not clear whether he obtains useful information for computing the secret keys. In this section, we show how to integrate distance bounding with known public key identification schemes such that no useful information is transferred.

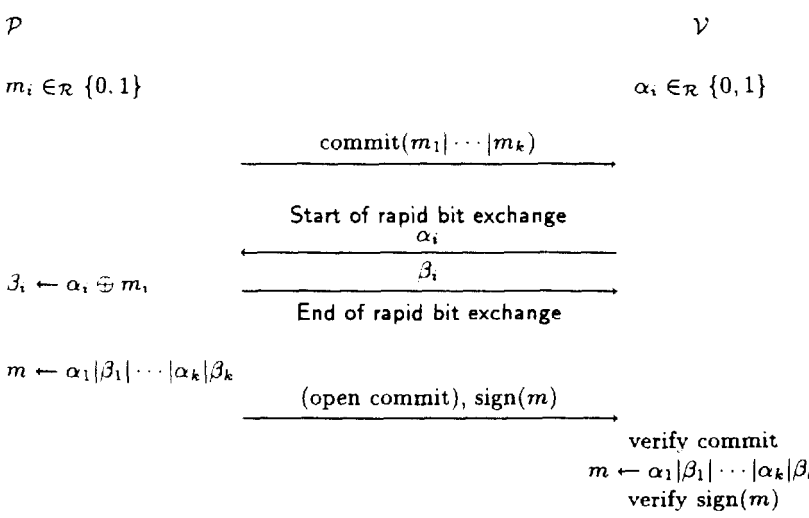

Fig. 4. Distance bounding to prevent both types of fraud.

3.1 Preventing mafia frauds

To prevent mafia frauds, we have the distance-bounding protocol dictate that $\mathcal{P}$ respond to challenges formed as the exclusive-or of the bits sent and received, instead of signing the concatenation. We illustrate this with the basic Fiat-Shamir scheme:

Step 1 $\mathcal{P}$ generates uniformly at random k numbers $R_i \in \mathbb{Z}_n^*$, and sends their squares R_i^2 mod n to $\mathcal{V}$. $\mathcal{P}$ also generates uniformly at random k bits β_i and commits to these bits (and their order) by sending a commitment on them to $\mathcal{V}$.

Step 2 $\mathcal{V}$ generates uniformly at random k bits α_i.

Step 3 Now the low-level distance-bounding exchanges can take place. Hereto, the following steps are repeated k times, for $i = 1, \ldots, k$.

- $\mathcal{V}$ sends bit α_i to $\mathcal{P}$.
- $\mathcal{P}$ sends bit β_i to $\mathcal{V}$ *immediately* after he receives α_i from $\mathcal{V}$.

Step 4 $\mathcal{P}$ opens the commitment on the bits β_i made in Step 1 by sending the appropriate information to $\mathcal{V}$. Furthermore, $\mathcal{P}$ determines the k responses $X^{c_i} R_i$ corresponding to challenges $c_i = \alpha_i \oplus \beta_i$, for $1 \leq i \leq k$, and sends them to $\mathcal{V}$.

$\mathcal{V}$ determines the k challenges c_i in the same way as $\mathcal{P}$ did, and verifies that the k responses are correct. Then $\mathcal{V}$ verifies whether the opening of the commitments by $\mathcal{P}$ is correct. If this holds, $\mathcal{V}$ computes an upper-bound on the distance to $\mathcal{P}$

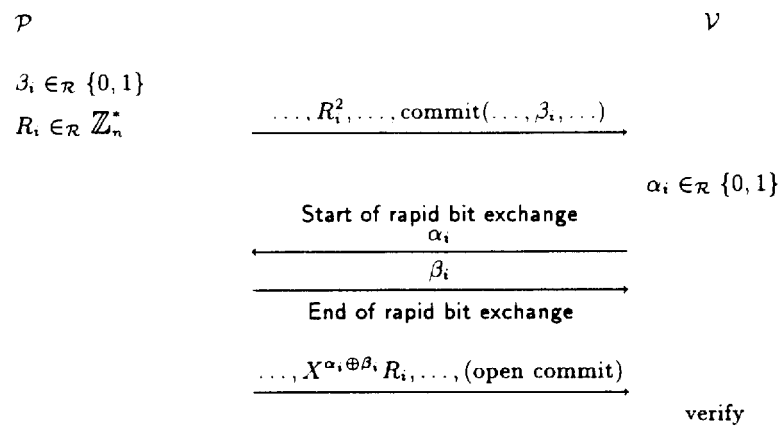

Fig. 5. Distance bounding in the Fiat-Shamir identification scheme.

using the maximum of the delay times between sending α_i and receiving β_i, for $i = 1, \ldots, k$. $\mathcal{V}$ accepts if and only if $\mathcal{P}$ is close by.

Proposition 2. *If the commitment scheme is secure, then this protocol is a proof of knowledge of a square root of* $X^2 \bmod n$ *that does not reveal any useful information for computing a root of* $X^2 \bmod n$.

Sketch of proof. In effect, this protocol is the parallel version of the basic Fiat-Shamir identification protocol. In [11] it is proven that this protocol reveals no useful information.

Since the binary challenges are chosen *mutually* random, the verifier cannot choose them as the outcome of a collision-free hash-function of the information known to him before Step 2. That is, the verifier does not receive information that he cannot compute himself. In particular, the transcript of an execution of the protocol cannot be used as a digital signature to convince others that the execution took place.

Proposition 3. *If the commitment scheme is secure,* $\mathcal{P}$ *is not close by to* $\mathcal{V}$ *and both follow the protocol, then the mafia fraud has probability of success at most* $1/2^k$.

Sketch of proof. In order to have any chance at all of having $\mathcal{V}$ accept, the fraudsters $\overline{\mathcal{P}}$ and $\overline{\mathcal{V}}$ must perform the rapid bit exchange first entirely with $\mathcal{V}$ and then with $\mathcal{P}$ (or vice versa), otherwise $\mathcal{V}$ will not accept because the computed upper-bound on the distance will not be tight enough (see Figure 6).

However, since a commitment was sent in Step 2, it is clear that with probability $1 - 1/2^k$ the fraudsters cannot prevent $\mathcal{P}$ and $\mathcal{V}$ from ending up with

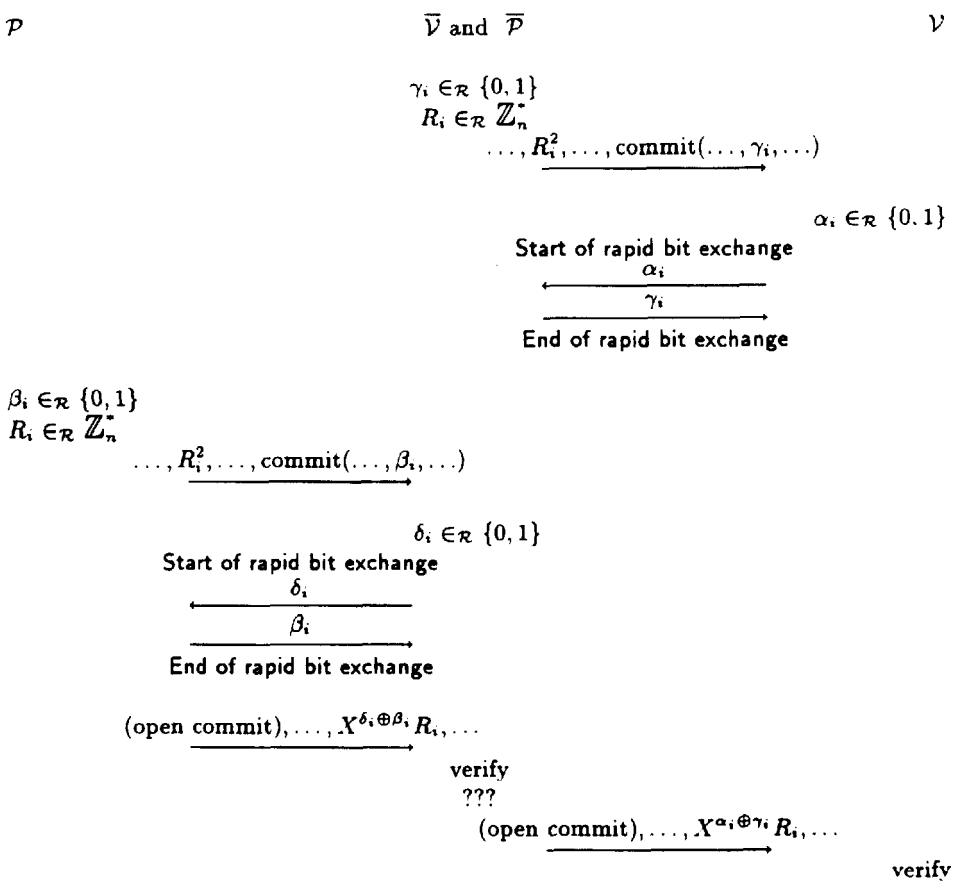

Fig. 6. Can $\overline{\mathcal{P}}$ and $\overline{\mathcal{V}}$ apply a mafia fraud?

at least one different challenge (i.e. $\beta_i \oplus \delta_i \neq \alpha_i \oplus \gamma_i$). Therefore, at least one response of $\mathcal{P}$ is correct with respect to a challenge that is complementary to the challenge $\mathcal{V}$ expects a response to. Clearly, one cannot convert $X^c R$ to $X^{c \oplus 1} R$ without knowing X.

Note that if at least one of $\mathcal{P}$ and $\mathcal{V}$ generates the challenge bits according to a distribution other than the uniform one (i.e., does not follow the protocol), this only increases the probability of successful cheating for $\overline{\mathcal{P}}$ and $\overline{\mathcal{V}}$.

Although we had the prover commit himself, it does not really matter whether the prover or the verifier commits. This holds for the protocol in the next section as well.

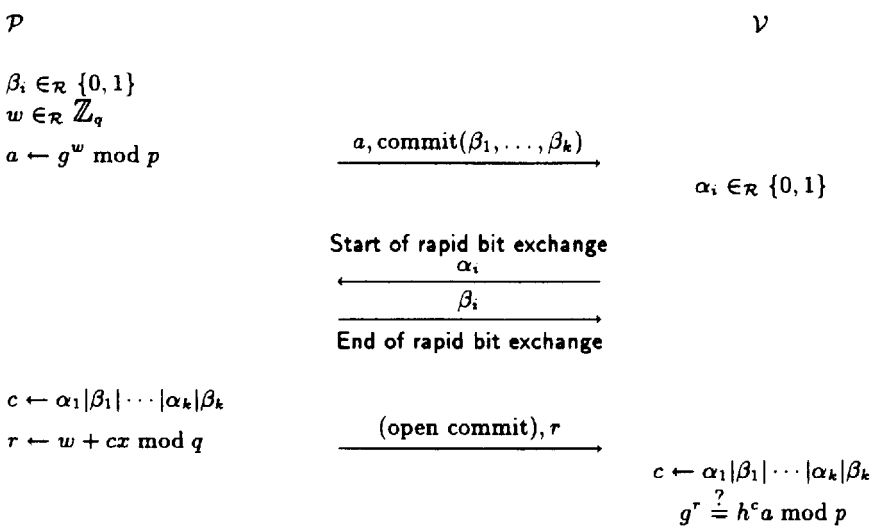

Fig. 7. Distance bounding in the Schnorr identification scheme.

3.2 Preventing both types of fraud

In order to prevent both types of frauds, as in Subsection 2.5, one can straightforwardly modify this protocol. To this end, in Step 1 $\mathcal{P}$ commits to k bits m_i, and in Step 3 $\mathcal{P}$ will reply with response bits $\beta_i = \alpha_i \oplus m_i$. Finally, the responses of $\mathcal{P}$ in Step 4 must be computed with respect to the multi-bit challenge $\alpha_1|\beta_1|\cdots|\alpha_k|\beta_k$ (using the concatenation of the xor-values does not prevent mafia frauds). This technique can be integrated in (minimum disclosure) identification schemes with multi-bit challenges, such as [5, 14, 15, 17], retaining the same security level for $\mathcal{P}$. Observe that the same propositions hold for this modified protocol; the only distinction is that the challenge bits *can* be chosen as the outcome of a collision-free hashfunction, and hence the transcript can serve as a digital signature.

Figure 7 shows how one might incorporate distance bounding into the Schnorr identification scheme. In this protocol, $(p, q, g, h = g^x \bmod p)$ is the public key of $\mathcal{P}$(as in [17]).

4 Distance bounding in wallets with observers

Up to now, we have considered distance-bounding protocols in a model with two legitimate parties. In this section, we will discuss distance bounding in a certain three-party setting. The goal of $\mathcal{V}$ is to determine an upper-bound to $\mathcal{P}$, and the task of the intermediary is to prevent undesired flow of information between

$\mathcal{P}$ and $\mathcal{V}$. Our technique allows the intermediary to prevent common coinflips between $\mathcal{V}$ and $\mathcal{P}$. This can be thought of as a generalization of the "warden's problem" (see [18]).

Recently, transaction systems based on "wallets with observers" have been proposed (see [6]). This setting can simultaneously offer privacy and security to an unprecedented extent. This is achieved by embedding within each user-module a tamper-resistant device called an observer. The observer is incorporated in a user-module in such a way that any message it sends to the outside world has to pass through the user-module. That is, the user-module acts as an intermediary party. The benefit of this setting is that one can design protocols such that the observer and the user-module both have to participate in order to have a verifier accept. In this way, a user cannot, say, double-spend the same coin in an electronic cash system since the observer will not participate a second time (see e.g. [3]).

Often, it will be sufficient to prevent outflow (any information going from the observer to the verifier not specified by the protocol) and inflow (any information going from the verifier to the observer not specified by the protocol). Inflow and in particular outflow can be a serious threat to the privacy of the user.

In [8] the privacy aspect of the wallet with observer setting has been investigated under an even more stringent requirement: even if an observer were to store all information it receives during the period it is embedded within a user-module, it still should be impossible (independent of computing resources) to link a payment to a user by examining afterwards the information inside the observer and all information gathered by the verifying parties. This possibility is not excluded by preventing inflow and inflow, since for example a single random number known to both an observer and a shop would enable linking: the fact that the user-module took part in generating it (so that no information could be encoded within it, thus preventing both inflow and outflow) is irrelevant in this matter. That is, one must also prevent "common coinflips." In [8], the term "shared information" is proposed, encompassing inflow, outflow, and common coinflips. The essential technique ("divertability") needed to prevent shared information in such a setting has been proposed earlier by Desmedt in [9], and was generalized in [16]. Prevention of shared information in some instances can be viewed as a slight generalization of divertability, in that the keys have to be shared together with the intermediary in a suitable way.

A fraud that can be applied in this three-party setting is one in which a user illegitimately uses an observer embedded within someone else's wallet. A possible motivation for doing so is that typically observers will gather (part of) negative credentials which can prevent the user from doing transactions he would like to do (see e.g. [7]). Also, another observer might have (part of) certain positive credentials the user would like to make use of. One can imagine a fraudulent organization specializing in lending, at a distance, observers with positive credentials (or without certain negative ones) to users who are willing to pay for this. In effect, when the user wants to do a transaction for which he needs certain positive credentials, he could use a radio link with the fraudulent organization

and lets an appropriate observer authorize the transaction. We will call this the "observer fraud."

4.1 Preventing the observer fraud

Using our distance-bounding technique we show how the verifier in the three-party setting can determine an upper-bound on the distance to the observer, such that the user-module can prevent shared information. We only describe one protocol that meets the most stringent requirements: no shared information, no release of useful information for computing the secret key, and the verifier obtains no information that he could not have computed himself (transcripts cannot serve as proof that the protocol took place). For easy comparison with the distance-bounding protocol shown in Section 3, our discussion will be based on the three party version of the Fiat-Shamir protocol (see [9, 16]).

We need a new notion called a "xor-commitment scheme." This is a commitment scheme which enables one to commit to the exclusive-or $\alpha \oplus \beta$ of two bits α and β, whereas one only knows a commitment on β but not β itself. In addition, one should be able to open the xor-commit if and only if one knows how to open the commitment on β, and this opening information must leak no Shannon information on the bits α and β, and the random choices involved in the commitment on β.

An implementation of an xor-commitment scheme can be realized with RSA, based on the technique of probabilistic encryption (see [13]). Let n be a Blum integer. In order to encrypt a bit α, the commiter chooses $r \in \mathbb{Z}_n^*$ at random and computes $\text{commit}(\alpha) := (-1)^\alpha r^2 \bmod n$. According to the quadratic residuocity assumption (see [13]), it is infeasible to decide whether $\text{commit}(\alpha)$ is a quadratic residue or not (i.e., whether $\alpha = 1$ or 0), unless one knows the factorization of n. Given a commitment $\text{commit}(\alpha) = (-1)^\alpha r^2 \bmod n$ of a bit α and a commitment $\text{commit}(\beta) = (-1)^\beta s^2 \bmod n$ of a bit β, it follows that $\text{commit}(\alpha \oplus \beta) = (-1)^{\alpha\oplus\beta} r^2 s^2 \bmod n$ is an xor-commitment on $\alpha \oplus \beta$. When opening this commit, one reveals $rs \bmod n$, which does not contain any information on s.

We denote the observer by $\mathcal{O}$, the verifier by $\mathcal{V}$ and the user-module by $\mathcal{U}$. For clarity, we leave out the fact that to prevent shared information the secret and public keys must be shared between the observer and the user-module in a suitable way. It is not hard to see how to do this using some of the techniques suggested in [7].

In the protocol, $\mathcal{O}$ knows a square root X of $X^2 \bmod n$, and $\mathcal{O}$ wishes to convince $\mathcal{V}$ of this fact in such a way that $\mathcal{U}$ does not learn it, whereas $\mathcal{U}$ can be ensured that there is no shared information. $\mathcal{V}$ wants to be convinced not only of the fact that $\mathcal{O}$ knows a square root of $X^2 \bmod n$, but also that $\mathcal{O}$ is close by. In essence, this is the setting of the mafia fraud, with the intermediary party ($\overline{\mathcal{P}}$ and $\overline{\mathcal{V}}$ in mafia frauds, and $\mathcal{U}$ in this situation) also trying to prevent shared information.

The protocol is as follows (see Figure 8):.

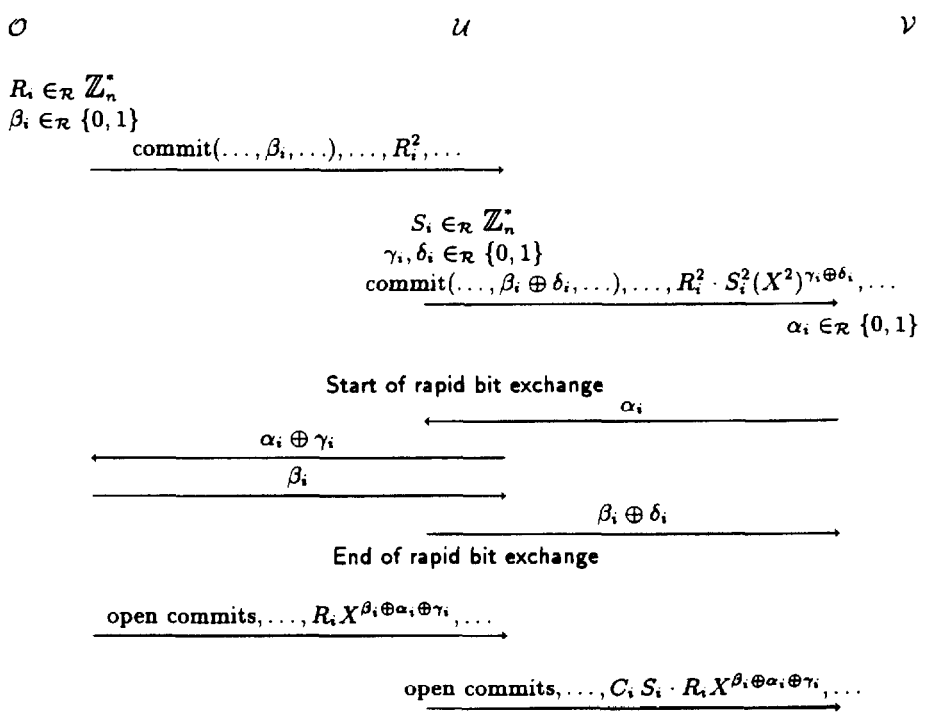

Fig. 8. Diverted Fiat-Shamir identification protocol with distance bounding.

Step 1 $\mathcal{O}$ generates k random numbers $R_i \in_{\mathcal{R}} \mathbb{Z}_n^*$ and sends the squares R_i^2 mod n of these numbers to $\mathcal{U}$. $\mathcal{O}$ also generates k bits β_i, and sends a xor-commitment on them to $\mathcal{U}$. (Clearly, if we use the specific xor-commitment just described, a commitment for each bit would be needed.)

Step 2 $\mathcal{U}$ first verifies that the numbers received from $\mathcal{O}$ all have Jacobi symbol 1. If this is the case, he generates at random k bits $\gamma_i \in_{\mathcal{R}} \{0,1\}$ as well as k bits $\delta_i \in_{\mathcal{R}} \{0,1\}$. $\mathcal{U}$ also generates k numbers $S_i \in_{\mathcal{R}} \mathbb{Z}_n^*$. He then computes the k products $R_i^2 \cdot S_i^2 (X^2)^{\gamma_i \oplus \delta_i}$ mod n and sends them to $\mathcal{V}$. $\mathcal{U}$ also sends xor-commitment(s) on $\beta_i \oplus \delta_i$ to $\mathcal{V}$.

Step 3 $\mathcal{V}$ generates k challenge bits $\alpha_i \in_{\mathcal{R}} \{0,1\}$, which he will use for the rapid bit exchange.

Step 4 Now the rapid exchange of bits can take place. Hereto, the following four exchanges are repeated k times, for $i = 1, \ldots, k$:

- $\mathcal{V}$ sends bit α_i to $\mathcal{U}$.
- $\mathcal{U}$ sends $\alpha_i \oplus \gamma_i$ to $\mathcal{O}$ *immediately* after receiving α_i.
- $\mathcal{O}$ sends challenge bit β_i to $\mathcal{U}$ *immediately* after receiving $\alpha_i \oplus \gamma_i$.

– $\mathcal{U}$ sends $\beta_i \oplus \delta_i$ to $\mathcal{V}$ *immediately* after receiving β_i.

Step 5 $\mathcal{O}$ opens the k commits on the bits β_i to $\mathcal{U}$ $\mathcal{O}$ also computes the k responses $R_i X^{\beta_i \oplus \alpha_i \oplus \gamma_i} \bmod n$ and sends them to $\mathcal{U}$.

Step 6 $\mathcal{U}$ verifies whether the responses of $\mathcal{O}$ are correct with respect to the challenges $\alpha_i \oplus \gamma \oplus \beta_i$ and the squares received from $\mathcal{O}$ in Step 3. $\mathcal{V}$ verifies whether the bits that $\mathcal{O}$ sent to him in Step 4 are those he commited to. If all the verifications hold, then $\mathcal{U}$ computes the k responses $C_i\, S_i \cdot R_i X^{\alpha_i \oplus \gamma_i \oplus \beta_i} \bmod n$ by multiplying, for $1 \leq i \leq k$, the i-th response of $\mathcal{O}$ by $S_i \bmod n$ and a correction-factor C_i. The correction-factor is equal to $X^2 \bmod n$ if and only if $\gamma_i \oplus \delta_i = 1$ and $\alpha_i \oplus \beta_i \oplus \delta_i = 1$, otherwise it is equal to 1. $\mathcal{U}$ sends all these responses to $\mathcal{V}$. Furthermore, $\mathcal{U}$ opens the xor-commitments to the k values $\beta_i \oplus \delta_i$ to $\mathcal{V}$.

Afterwards, $\mathcal{V}$ verifies whether the responses of $\mathcal{U}$ are correct with respect to the challenges $\beta_i \oplus \delta_i \oplus \alpha_i$ and the squares received from $\mathcal{V}$ in Step 3. He also verifies whether the bits received from $\mathcal{U}$ in Step 4 are those $\mathcal{U}$ commited to. If all the verifications hold, then $\mathcal{V}$ derives an upper-bound on the physical distance to $\mathcal{O}$ by using the maximum of the delays between sending out α_i and receiving $\beta_i \oplus \delta_i$ from $\mathcal{U}$, for $1 \leq i \leq k$. $\mathcal{V}$ accepts if and only if $\mathcal{O}$ is close by.

Although we write commit$(\ldots, \beta_i, \ldots,)$ we do not mean to imply with this that a multi-bit commitment must necessarily be used: one might as well use k single-bit commitments.

It is straightforward to show that $\mathcal{V}$ accepts if all parties follow the protocol, and that Propositions 2 and 3 hold.

Since one can easily show that for each view of $\mathcal{V}$ and for each view of $\mathcal{O}$ in this protocol, there is exactly one set of random choices that could have been made by $\mathcal{U}$ such that the views are from the same execution of the protocol, there is no shared information. Clearly, for the protocol as we described it, this only holds for executions concerning proof of knowledge of the particular number $X^2 \bmod n$. However, as we noted before, if the knowledge of $X^2 \bmod n$ is divided between $\mathcal{O}$ and $\mathcal{U}$ in a suitable way (as described in [8]), the property of absence of shared information holds for the set of *all* proofs of knowledge, regardless of the particular number $X^2 \bmod n$ that the proof is concerned with.

Finally, as in Proposition 3 it is easy to see that the following must hold.

Proposition 4. *If $\mathcal{O}$ and $\mathcal{V}$ follow the protocol, then $\mathcal{U}$ cannot (with probability of success greater than $1/2^k$) trick $\mathcal{V}$ into believing that $\mathcal{O}$ is close by if this is not the case.*

As before, if at least one of $\mathcal{O}$ and $\mathcal{V}$ generates challenge bits according to a distribution other than the uniform one, then $\mathcal{U}$'s probability of successful cheating will only increase.

5 Open problems and further work

We would like to present two potentially fruitful areas for further investigation.

First is the physics and practical implementation of distance-bounding technology. We know little about the physical limits or precisely how best to use current technology. Some experimental work might also be interesting.

Second is dealing with a problem not adressed here. The techniques presented do not prevent frauds in which a distant party with access to the secret keys is *cooperating* with a party close by (without conveying the secret keys). The frauds were suggested informally under the name "the terrorist fraud" by Desmedt in [9]. We are currently working on some ideas preventing such frauds using distance bounding.

Acknowledgements

We are grateful to Niels Ferguson for some suggestions concerning distance bounding in wallets with observers, to Torben Pedersen for the particular realization we described of an xor-commitment, and to Corné Hoogendijk for discussions on the physical realizations of distance-bounding.

References

1. Bengio, S., Brassard, G., Desmedt, G., Goutier, C. and Quisquater, J., "Secure Implementation of identification schemes," Journal of Cryptology, 4 (1991), pages 175–183.
2. Beth, T. and Desmedt, Y., "Identification tokens–or: solving the chess grandmaster problem," Crypto '90, Lecture Notes in Computer Science, Springer-Verlag (1991), pages 169–176.
3. Brands, S., "An efficient off-line electronic cash system based on the representation problem," C.W.I. Technical Report CS - T9323, april 1993, The Netherlands.
4. Brassard, G., Chaum, D. and Crépeau, "Minimum Disclosure Proofs of Knowledge," Journal of Computer and System Sciences, Vol. 37 (1988), pages 156–189.
5. Brickell, E. and McCurley, K., "An interactive identification scheme based on discrete logarithms and factoring," Journal of Cryptology, Vol. 5, no. 1 (1992), pages 29–39.
6. Chaum, D., "Achieving electronic privacy," Scientific American, Aug. 1992, pages 96–101.
7. Chaum, D. and Pedersen, T., "Wallet Databases with Observers," Proceedings of Crypto '92, Abstracts, Santa Barbara, August 1992, pp. 3.1-3.6.
8. Cramer, R. and Pedersen, T., "Improved privacy in wallets with observers," in: these proceedings.
9. Desmedt, Y., "Major security problems with the 'unforgeable' (Feige)-Fiat-Shamir proofs of identity and how to overcome them," SecuriCom '88, SEDEP Paris, (1988), pages 15–17.
10. Desmedt, Y., Goutier, C., and Bengio, S., "Special uses and abuses of the Fiat-Shamir passport protocol," Crypto '87, LNCS 293, Springer-Verlag (1988), pages 16-20.
11. Feige, U., Fiat, A. and Shamir, A., "Zero-knowledge proofs of identity," Journal of Cryptology 1 (1988), pages 77–94.

12. Fiat, A. and Shamir, A., "How to prove yourself: practical solutions to identification and signature problems," Crypto '86, Springer-Verlag, (1987), pages 186–194.
13. Goldwasser, S. and Micali, S., "Probabilistic Encryption," Journal of Computer and System Sciences, Vol. 28 (1984), pages 270–299.
14. Guillou, L. and Quisquater, J.-J., "A 'paradoxical' identity-based signature scheme resulting from zero-knowledge," Crypto '88, Springer-Verlag, pages 216-231.
15. Okamoto, T., "Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes," Proceeding of Crypto '92, pages (1-15) - (1-25).
16. Okamoto, T. and Ohta, K., "Divertible zero knowledge interactive proofs and commutative random self-reducibility," Eurocrypt '89, Springer-Verlag, pages 134-149.
17. Schnorr, C.P., "Efficient Signature Generation by Smart Cards," Journal of Cryptology, Vol. 4, No. 3, (1991), pages 161–174.
18. Simmons, G., "The prisoner's problem and the subliminal channel," Crypto '83, Santa Barbera (1983), Plenum, New York, pages 51–67.