WEAKNESSES OF UNDENIABLE SIGNATURE SCHEMES

(Extended Abstract)

Yvo Desmedt*
Dept. of EE & CS
P.O.Box 784
Milwaukee, WI 53201
U.S.A.

Moti Yung
IBM T. J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598
U.S.A.

Abstract

The nice concept of undeniable signatures was presented by Chaum and van Antwerpen [10]. In [7] Chaum mentioned that "with undeniable signatures only paying customers are able to verify the signature." Using methods based on "divertible zero-knowledge proofs" and "distributed secure *mental* games played among cooperating users", we show that in certain contexts *non*-paying verifiers can check the signature as well, thus demonstrating that the applicability of undeniable signatures is somewhat restricted and must rely on the physical (or other) isolation of the verifying customer. In addition, we show that the first undeniable signature schemes suffer from certain security problems due to their multiplicative nature (similar to problems the RSA signature scheme has).

1 Introduction

Undeniable signatures were introduced in [10], further work on the subject is given in [7, 21, 2]. Unlike digital signatures, undeniable signatures cannot be verified without cooperation of the signer. This means that in an initial "commitment phase" the signer sends a message together with a commitment information.

*This research has been partially supported by NSF Grant NCR-9004879.

D.W. Davies (Ed.): Advances in Cryptology - EUROCRYPT '91, LNCS 547, pp. 205-220, 1991.

Later (*e.g.*, one year later), in the "verification phase", the signer will prove that this commitment corresponds to a signature; such verification proofs can be zero-knowledge as presented at Eurocrypt '90 [7]. Chaum [7] (and later also in [2]) mentioned that "undeniable signatures are preferable to digital signatures for many upcoming applications." The following (and other) applications which exemplify the potential of the notion of "undeniable signature" were given:

- "Consider ... the signature a software supplier issues on its software, which allows customers to check that the software is genuine and unmodified. With undeniable signatures, *only paying customers are able to verify the signature*, and they are still ensured that the supplier is accountable for the software.

- "All manner of inter-organizational messages ... are a natural candidate for signatures that provide for dispute resolution. But self-authentication would greatly increase the illicit salability of such information."

In this paper we demonstrate that, in fact, the signer in the verification phase *cannot* restrict the recipients of his proof of signature validity in scenarios where the set of users can communicate with each other (such as in public networks). In other words, while the prover thinks that he is proving the validity of his signature to a specific person, he could *without his knowledge be proving it to a large group of people, convincing all of them simultaneously.* Thus, in the case of signing software releases, a dishonest customer could buy the software and sell it to a group of users at half the price. Then, when these customers want to check the validity of the signature the customer will help the others in checking the validity of their copies, as we will discuss in Section 3. Observe that when the group of users is afraid that the above customer is a crook (a computer hacker), they can still be convinced of the validity (invalidity) of the software release.[1] In addition, we show how to use divertible zero-knowledge to attack the original undeniable signature scheme, in which one of the verifiers can be fooled to believe he is running a legal protocol, while actually he is talking with an intermediate cheating party.

[1] Due to initial remarks of [5] following the initial presentation of these paper's ideas, and in order to clarify any possible confusion, we will discuss this last aspect in sufficient details in Section 3.2.

In the final version of this paper we will present certain settings in which under additional assumptions about the context, the problems presented can be reduced.

The first undeniable signatures schemes suffer from similar problems as the RSA signature scheme does. While it was proven secure with respect to key-only attacks[2] [7], an eavesdropper/ active eavesdropper can, during the commitment phase, modify commitments for signatures into ones for other messages. In particular, the scheme is insecure with respect to an "existential chosen plaintext attack": agreeing to sign and verify a randomly looking message chosen by the verifier, may imply that the signer is actually committed to another "meaningful message he has no information about". In Section 4 we explain how this and more can be achieved.

We first overview Chaum's zero-knowledge undeniable signature scheme.

2 The undeniable signature scheme

We review the undeniable signature scheme in which all users know G and g, where G is a group which order is p, a prime, and g is a generator. Each user announces g^x as public key and keeps his own x secret. To commit to a signature for the message m, the sender sends: (m, m^x) in the commitment phase. Let us denote $z = m^x$.

When asked to validate the signature (during the verification phase) the following *confirmation* protocol[3] is executed as in Figure 1.
When the receiver's checking returns correctly, he accepts the confirmation of the message as being valid. Observe that if the *signer has committed to more than one message the verifier must provide* m.

Chaum also discusses a *disavowal* protocol. If the receiver has received for the message $m \neq 1$ a commitment $z = m^{x'}$, (note that each element of the group, can be written in such a form because the order of the group is a prime, so each element but the identity is a generator) where $x' \neq x$, then the sender can prove that z is not of the proper form. Since this protocol is almost irrelevant in our context we do *not* discuss it in detail.

[2] when the attacker tries to forge, solely based on the availability of the public keys.

[3] Choosing a in the set A with uniform probability distribution and independently of other events is denote as $a \in_R A$.

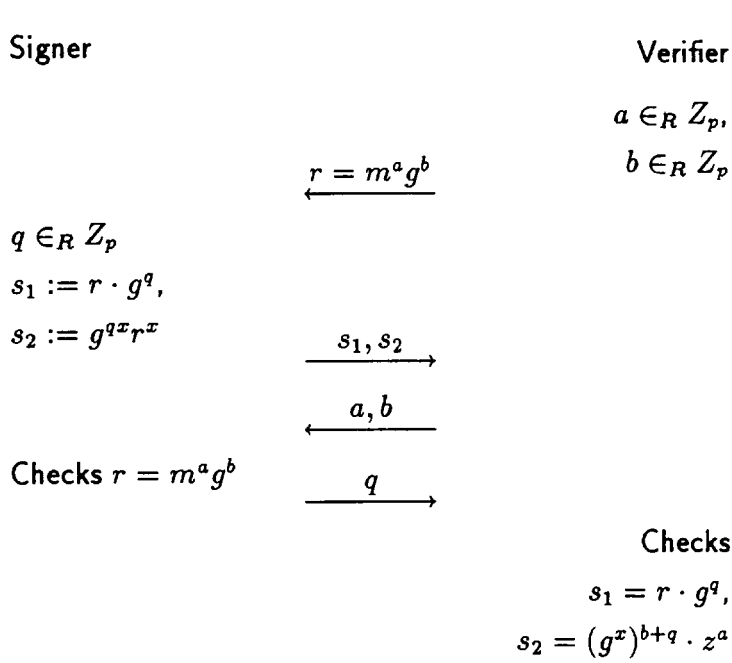

Figure 1: Chaum's zero-knowledge confirmation scheme.

The scheme works since for a forger, using the public-key directory (the sender's public-key), the probability of coming up with a pair which is a message and a commitment to its signature and cannot be successfully disavowed is negligible for large enough values of p (assuming discrete logarithm mod p is hard). So, the scheme is secure with respect to key-only attack.

3 Verification by multiple unknown verifiers is possible

In this section we show that in *any* undeniable signature scheme the signer has no control on how many verifiers he confirms the validity of the signature to. To this end, the verifiers collaborate using the concept of secure function evaluations

knows also as "Mental Games" [20, 19, 9]. Mental games allow n players to play any partial-information game over the telephone (using a conference call) following the specified rules of the game such that no one can cheat, assuming that half of the players are honest. (The last result even guarantees that one player in the game is unconditionally secure). (The game can be played over secure physical lines assuming two thirds are honest as, for example, in [8]). It enables secure distributed computing of a function where the players compute together the result of a function correctly, based on their private inputs while maintaining the secrecy of their inputs.

In the honest case the sender validates the signature to a verifier. Now, we replace this one verifier by a group of verifiers. The rules of the game are that *all* the verifiers get convinced of the validity of the commitment once the protocol is finished. We now distinguish two cases: the group of users trust that one individual, called the trusted party, will not impersonate the signer or will attempt to give fraudulent zero-knowledge proofs. In the second case we do not make this assumption.

3.1 All verifiers trust one of them

This simpler scenario was raised in [5] and is similar to a possible scenario mentioned in [7] (see remark 1 at the end of this section). It is actually a methodological way to approach the general setting of the next subsection. In this case we do not have to use the full power of the mental games and the protocol is very simple. The protocol runs in the background by the collaborating parties and relies on zero-knowledge and bit-commitment protocols.

The technical details, in general, are as following. The collaborating verifiers generate a random string of bits to be used in the validation protocol (for example, they generate the required bits for each step) executed with the sender. To do this, the verifier's work together in a sub-protocol hidden from the sender. The verifiers all commit to random bits (using the simple bit commitment protocol due to Blum [1], they can initially commit to many bits and then open them as needed and fast). When they open these commitments these private bits are exclusive-ored together. This gives the result of the sub-protocol which is a common string of random bits to be used as the randomness in the interactive step with the sender in the validation stage. This randomness is trusted by

all verifiers, even when they do not trust each other. Thus, this random string is fed into a trusted box which executes the protocol as an actual verifier with the sender and all verifiers having access to the transcript of the validation are simultaneously convinced. The on-line opening of commitments and ex-oring is a very simple computation, thus this scenario is important and enables one trusted simple device to serve many verifiers at the same time (while paying only once) even when the random number generator in that simple device and many of the verifiers are possibly unreliable or predictable.

Even when the sender requires that the verifier (to whom, for example, he sold his software release) uses some adaptation such as (for example) forcing the verifier to use a public key to encrypt a message, all the other verifiers can be convinced even without learning the key used (since instead of opening the key, the actual verifier can convince the rest of the players in a zero-knowledge fashion that the key was used to encrypt data based on their common random data, this validation proof can even be executed in an off-line fashion after the completion of the protocol).

As a conclusion, one can see that in an environment where the verifier is not physically isolated, it is always possible for him (using the sender) to convince others of the validity of a signature, without assuming that the others "fully trust" the actual verifier (this opens a possibility of "validation piracy").

3.2 No verifier is trusted by the others

We now discuss the case that no verifier is trusted and that verifiers are afraid that the one who is communicating with the prover (signer) will reveal something to him to allow him to give a fraudulent proof. More they are afraid that one verifier himself will attempt to give a false proof to all other verifiers. We can view this case as a fault-tolerant extension of the previous case, and in this case the verifiers use the full power of mental games.

Mental games allow many individuals to securely compute the output of a multiple input algorithm such that none of the other individuals will know the inputs used by the others. It has been proven that any such algorithm can be securely executed (guaranteeing privacy of inputs) provided that half of the participants are honest [20, 19, 9]. So the following holds:

Theorem 1 *No undeniable signature scheme is secure against a multi-verifier*

attack provided that half of the verifiers are honest.

Proof. The proof relies on [20]. Suppose that a confirmation protocol of an undeniable signature scheme, in which there is one prover and one verifier, is given. We call this scheme "the game". We now modify the game into what we call the "modified game". Assume that the verifier has to compute (at some stage) some $A(x, y, s)$ (where x is public, y is an information which is secret, but $E(y)$, is a public encryption of y (for example a signature or an encryption key) which is encrypted in public so that validation based on y can be executed, and s is secret but known to the verifier), and send the result to the prover. Then we modify this into a computation of $A'(x, y, s_1, s_2, \ldots, s_m)$, where m is the number of multi-verifiers and s_i is secret but known to verifier i. Now this A' is computed using the concept of mental game and the output is sent to the prover. All the verifiers are convinced during the game that indeed the right y key corresponding to $E(y)$ is used as an additional input of the actual verifier. For example, in the case that s corresponds to a (uniformly) randomly chosen string, $A'(x, y, s_1, s_2, \ldots, s_m) = A(x, y, s_1 \oplus s_2 \oplus \cdots \oplus s_m)$. (A more formal proof will be given in the final paper.) □

It is clear from the proof that the approach in this case is almost identical to the one given in Section 3.1, but that the calculation of what has to be sent to the prover is done using mental games. Recently Ohta–Okamoto–Fujioka have given as open problem the question of the existence of "an equivalent condition that plural verifiers can not be convinced of the validity of a signature." The above answers in the negative this question, (and thus, physical assumptions such as isolation or other contextual constraints are necessary).

3.3 Using divertible zero-knowledge

Now, we discuss a worse scenario which applies to undeniable signatures of the type of [7]. Let us illustrate the scenario. Alice wants to buy a nice software release, but she is a software pirate (with a scul of a businesswoman). She convinces Bob that she is a representative of the software company, so Bob pays her to buy the software. Alice does this, and then continues to sell it to colleague pirates, without paying the company. When Bob wants his software validated he plays the confirmation protocol with Alice. Alice then plays *simultaneously* the

protocol with the software company. She may also, at the same time, convince all her colleagues of the validity of her software, for reduced fees.

Figure 2 explains the technical details, which are based on "divertible zero-knowledge proofs" [16, 23] where the verifier in the middle diverts the exchanged messages.

Observe that at the end both verifiers are completely convinced of the validity of the commitment.

The verifier in the above protocol can be replaced by a chain of verifiers in which Alice, in the above example, is communicating with the sender, the other cooperating pirates are in the middle between Alice and Bob, and Bob (the victimized customer) is at the end of the chain. The cooperating (but not trusting) colleagues in the middle are all (but the honest Bob) aware of the diversion and are taking part in it (multi-step diversion, each diverting the previously given information). Once the protocol is successfully terminated, they are all simultaneously convinced.

Remark 1 David Chaum has communicated to us the following three points [6] which we present (based on our understanding). The first point is that the exact use of mental games, mentioned in the pre-proceedings version of this work, was not clear; we hope the above clarifies it. Second, he pointed out that there are other means of isolation of the verifier (rather then only physical) which help in prevention of collaborations, he suggests exploiting the fact that, currently, mental games require a certain amount of computational time, and thus, imposing temporal restriction on the verifier's responses may effectively isolate the verifier. This was discussed in [5] and for the technique and details see Chaum's paper based on his presentation. Finally, Chaum's third point is that he was aware of possible covert cooperation of many verifiers (perhaps similar to the case of subsection 3.1) and had mentioned in the *recent works* section of his Eurocrypt-90 paper [7] (on page 463) a solution to such attacks by applying a *verifier commit protocol* (which may possibly be the protocol relying on temporal constraints which was suggested in the discussion of the previous point).

We would like to say that it is only natural (in a cryptologic setting) that we try to point out a broad range of weaknesses while David tries to point out as broader as possible scenario in which the weaknesses do not apply. We view the remarks and the discussion as a healthy exchange and thank him for his remarks.

Sender		Verifier		Victimized Verifier
				$a, b \in_R Z_p$
		$a', b' \in_R Z_p$	$\xleftarrow{r}$	$r := m^a g^b$
$q \in_R Z_p$	$\xleftarrow{\tilde{r}}$	$\tilde{r} := m^{a'} g^{b'} r$		
$s_1 := \tilde{r} \cdot g^q$, $s_2 := g^{qx} \tilde{r}^x$	$\xrightarrow{s_1, s_2}$			
		$\tilde{s}_1 := s_1/(m^{a'} g^{b'})$ $\tilde{s}_2 := s_2/(z^{a'}(g^x)^{b'})$	$\xrightarrow{\tilde{s}_1, \tilde{s}_2}$	
			$\xleftarrow{a, b}$	
		$\tilde{a} := a + a'$ $\tilde{b} := b + b'$		
	$\xleftarrow{\tilde{a}, \tilde{b}}$			
Checks $\tilde{r} = m^{\tilde{a}} g^{\tilde{b}}$	$\xrightarrow{q}$			
		Checks $s_1 = \tilde{r} \cdot g^q$, $s_2 =$ $(g^x)^{b+b'+q} \cdot z^{a+a'}$	$\xrightarrow{q}$	
				Checks $\tilde{s}_1 = r \cdot g^q$, $\tilde{s}_2 = (g^x)^{b+q} \cdot z^a$

Figure 2: Attack based on divertible zero-knowledge

(We note that, naturally, we are solely responsible for the writing of the above remark).

4 Vulnerability to on-line multiplicative attacks

In this section we assume that an *active* eavesdropper, Eve, is able to interfere during the commitment and possibly during the verification phase as well. As usual the active eavesdropper is located between the sender and the receiver. Such on-line security problems have been studied in completely different contexts in [18]. The attacks apply to the protocols in [10, 7].
We next present three attacks:

- A "known-plaintext attack" which can result in a commitment of the sender to a random message.
- A "chosen-plaintext attack" which can cause a commitment to a message which at the time of commitment, the sender has no knowledge about.
- An active relay "meddler attack" which causes the receiver to get a commitment and verification of one message, and the sender to commit and verify a totally different message.

Let $m_1, \ldots, m_c$ be the messages to which the signer has already committed himself and let the corresponding commitments be $z_1, \ldots, z_c$, which where eavesdropped by Eve. When, during the commitment phase, the signer sends m and $z = m^x$, Eve will modify the message into:

$$\tilde{m} = m^f \cdot g^{e_0} \cdot \prod_{i=1}^{c} m_i^{e_i}$$

and the commitment into:

$$\tilde{z} = z^f \cdot (g^x)^{e_0} \cdot \prod_{i=1}^{c} z_i^{e_i}$$

by choosing (arbitrary) f, e_i, and sending $\tilde{m}, \tilde{z}$ to the receiver. Observe that if all the z_i and z were proper commitments for m_i and respectively m, then $\tilde{z}$ is a

valid commitment for $\tilde{m}$. When the verification phase starts and the receiver (of the commitment) sends $\tilde{m}$ and $\tilde{z}$ for validation, Eve will forward those. At this stage the sender of the commitment is in a position that he could determine that he never committed to this message, *i.e.*, by having stored all the messages to which he ever committed. However, the disavowal protocol does not allow him to deny having committed to this signature, which is due to the above observation. (Recall that when m and z are given, such that $z = m^x$, the sender cannot execute a disavowal protocol for the fact of not sending m.) So no choice is left for him but to participate in the confirmation protocol and there is no need for Eve to interact in this protocol. This is a known plaintext attack where Eve is able to generate a set of random signatures.

Now we allow the m_i to be chosen by Eve, instead of by the sender. This is the chosen plaintext attack. This attack is very similar to the chosen plaintext attacks proposed against RSA. Similar techniques as discussed in [11, 14, 15, 12, 17, 13, 22] can also be used here for this purpose. The chosen plaintext attacks allow Eve to generate commitments for any message of her choice, by adapting her attack to this message and the signer has no information about the message he is committing himself to (since the message signed directly is random). It was noticed by Chaum that "blinding" is possible in the setting of the undeniable signature protocol, which implies that random messages will be signed. In fact, the attack above exploits exactly the possibility of "blinding" which is very dangerous (a double-ended sword) in this setting, once it is combined with chosen-plaintext.

We now present a variant of our attack which we call the "meddler attack". It applies to the protocol in [7]. In this new attack Eve will actively (with the help of the sender) convince the receiver that $\hat{m}$ is a valid message in such a way that the sender does not know $\hat{m}$! During the commitment phase Eve acts as a meddler and replaces m and z respectively by $\hat{m} = m^f \cdot g^e$ and $\hat{z} = z^f \cdot (g^x)^e$. When the verification phase starts and the receiver sends $\hat{m}$ and $\hat{z}$ to the sender, Eve will replace them by m and z. So the sender "believes" that he is confirming the validity of z as an undeniable signature for m, but he will (due to Eve) in fact convince the receiver that $\hat{z}$ is a valid undeniable signature for $\hat{m}$. The sender serves as an oracle for Eve to compute a commitment for one (say, what he believes to be a randomly looking) message, while in the process the sender commits to a totally different (possibly meaningful and harmful) message. Eve's

interaction is explained in Figure 3.

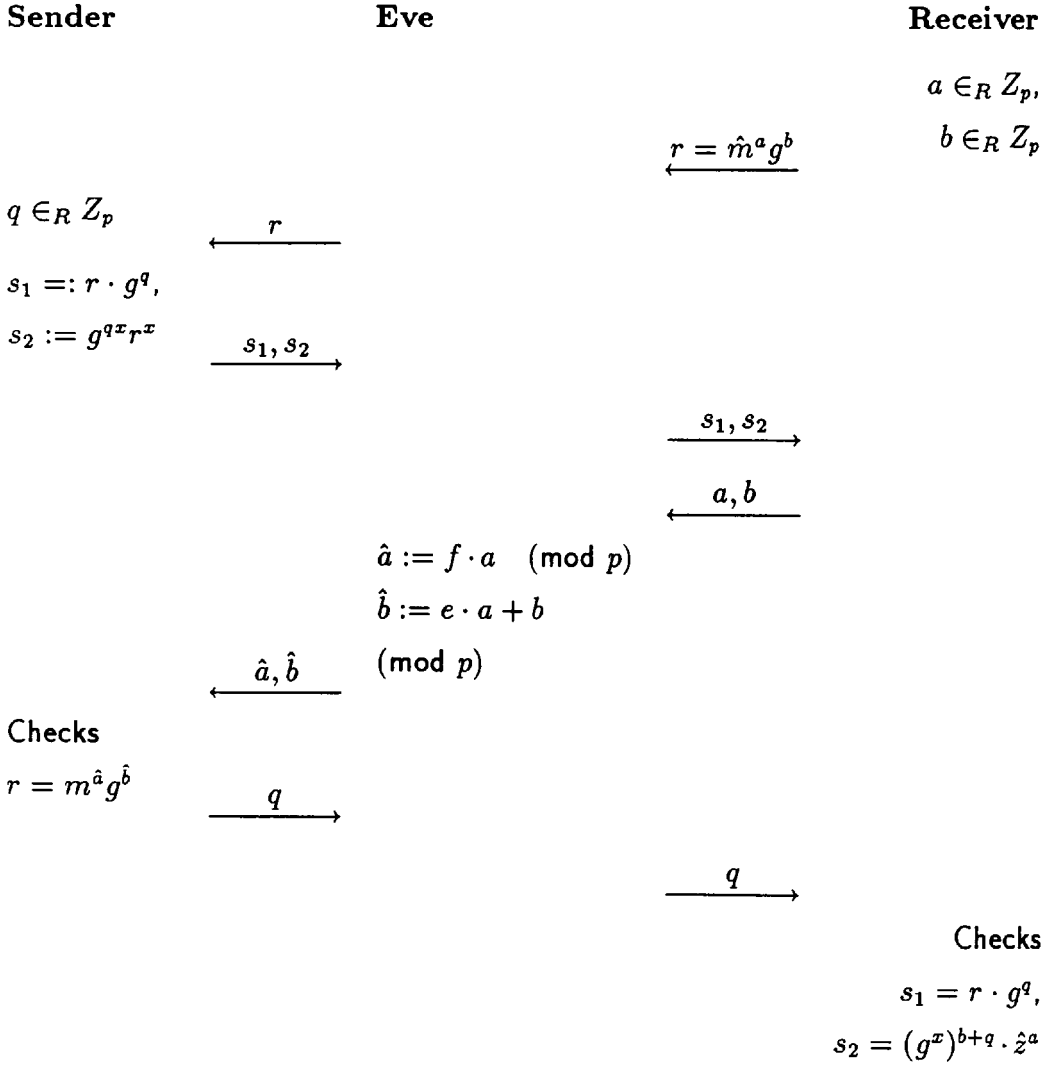

Figure 3: The meddler's, Eve's, interaction

5 Conclusions

We have presented certain scenarios in which carefulness is required when applying undeniable signatures. As with any other cryptographic primitive it is important to clarify and better understand the exact setting in which undeniable signature applies.

First we have observed that a verifier who behaves as an active relay-station provides an anonymous way of verifying the validity of the commitment to a multitude of verifiers. This demonstrates that the concept of anonymity, studied by Chaum [3, 4] is indeed very powerful and can also be used for cryptanalytic purposes. In particular, in protocols like "undeniable signature" where the protocol goals includes restriction to a "specified receiver", anonymous channels violate the goals and should be detected. Even when the channel is not anonymous, but relies on information which the verifier is committed to (for possible verification by a judge), exclusive use of the channel cannot be assured. Mental games play an important role in this context.

Secondly, we have demonstrated that multiplicative undeniable signature schemes suffer from weaknesses similar in nature to the RSA signature scheme. The proven secure non-multiplicative versions [21, 2] of the notion do not suffer from this disadvantage.

In the final version we will explain how the problem of multitude of unknown verifiers can be reduced by providing personalized commitments to signatures. This applies to such applications such as software validation, in settings where there exists an active authority that probes the software users.

Acknowledgement

We acknowledge Mike Burmester for having presented this paper and for useful discussions, Gus Simmons for his comments, and David Chaum for his remarks.

REFERENCES

[1] M. Blum. Coin flipping by telephone — a protocol for solving impossible problems. In *digest of papers COMPCON82*, pp. 133–137. IEEE Computer Society, February 1982.

[2] J. Boyar, D. Chaum, I. Damgard, and T. Pedersen. Convertible undeniable signatures. Presented at Crypto '90, August 12–15, 1990, Santa Barbara, California, U.S.A., to appear in: Advances in Cryptology. Proc. of Crypto '90 (Lecture Notes in Computer Science), Springer-Verlag, 1990.

[3] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2), pp. 84–88, February 1981.

[4] D. Chaum. The dining cryptographers problem: unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1), pp. 65–75, 1988.

[5] D. Chaum. On weaknesses of 'weaknesses of undeniable signatures'. Presented at the rump session of Eurocrypt '91, Brighton, U.K., April (Communicated to us by Gus Simmons.) 1991.

[6] D. Chaum. Personal Communication (over the phone, no coin flipping!).

[7] D. Chaum. Zero-knowledge undeniable signatures. In I. Damgård, editor, *Advances in Cryptology, Proc. of Eurocrypt '90 (Lecture Notes in Computer Science 473)*, pp. 458–464. Springer-Verlag, 1991. Åarhus, Denmark, May 21–24.

[8] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *Proceedings of the twentieth annual ACM Symp. Theory of Computing, STOC*, pp. 11–19, May 2–4, 1988.

[9] D. Chaum, I. Damgård, and J. van de Graaf. Multiparty computations ensuring privacy of each party's input and correctness of the result. In C. Pomerance, editor, *Advances in Cryptology, Proc. of Crypto '87 (Lecture Notes in Computer Science 293)*, pp. 87–119. Springer-Verlag, 1988. Santa Barbara, Ca., August 16-20, 1987.

[10] D. Chaum and H. van Antwerpen. Undeniable signatures. In G. Brassard, editor, *Advances in Cryptology — Crypto '89, Proceedings (Lecture Notes in Computer Science 435)*, pp. 212–216. Springer-Verlag, 1990. Santa Barbara, California, U.S.A., August 20–24.

[11] G. I. Davida. Chosen signature cryptanalysis of the RSA (MIT) public key cryptosystem. Tech. Report TR-CS-82-2, University of Wisconsin-Milwaukee, October 1982.

[12] W. de Jonge and D. Chaum. Attacks on some RSA signatures. In *Advances in Cryptology: Crypto '85, Proceedings (Lecture Notes in Computer Science 218)*, pp. 18–27. Springer-Verlag, New York, 1986. Santa Barbara, California, U.S.A., August 18–22, 1985.

[13] W. de Jonge and D. Chaum. Some variations on RSA signatures & their security. In A. Odlyzko, editor, *Advances in Cryptology, Proc. of Crypto'86 (Lecture Notes in Computer Science 263)*, pp. 49–59. Springer-Verlag, 1987. Santa Barbara, California, U. S. A., August 11–15.

[14] R. A. DeMilo, and M. J. Merritt Chosen signature cryptanalysis of public key cryptosystems. Technical Memorandum, Georgia Institute of Technology, October 1982.

[15] D. E. R. Denning. Digital signatures with RSA and other public-key cryptosystems. *Comm. ACM 27*, pp. 388–392, 1984.

[16] Y. Desmedt, C. Goutier, and S. Bengio. Special uses and abuses of the Fiat-Shamir passport protocol. In C. Pomerance, editor, *Advances in Cryptology, Proc. of Crypto '87 (Lecture Notes in Computer Science 293)*, pp. 21–39. Springer-Verlag, 1988. Santa Barbara, California, U.S.A., August 16–20.

[17] Y. Desmedt and A. Odlyzko. A chosen text attack on the RSA cryptosystem and some discrete logarithm schemes. In Hugh C. Williams, editor, *Advances in Cryptology: Crypto '85, Proceedings (Lecture Notes in Computer Science 218)*, pp. 516–522. Springer-Verlag, 1986. Santa Barbara, California, U.S.A., August 18–20.

[18] O. Dolev and A. Yao. On the security of public key cryptography. *IEEE Trans. Inform. Theory*, 29, pp. 198–208, March 1983.

[19] Z. Galil, S. Haber, and M. Yung. Cryptographic computations: secure fault-tolerant protocols and the public-key model In C. Pomerance, editor,

Advances in Cryptology, Proc. of Crypto '87 (Lecture Notes in Computer Science 293), pp. 135–155. Springer-Verlag, 1988. Santa Barbara, Ca., August 16-20, 1987.

[20] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth annual ACM Symp. Theory of Computing, STOC*, pp. 218–229, May 25–27, 1987.

[21] S. Micali. Public announcement at Crypto '89.

[22] J. H. Moore. Protocol failures in cryptosystems. *Proc. IEEE*, 76(5), pp. 594–602, May 1988.

[23] T. Okamoto and K. Ohta. Divertible zero knowledge interactive proofs and commutative random self-reducibility. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology, Proc. of Eurocrypt '89 (Lecture Notes in Computer Science 434)*, pp. 134–149. Springer-Verlag, 1990. Houthalen, Belgium, April 10–13.