Anonymity and Denial of Undeniable and Confirmer Signatures

Steven D. Galbraith[1], Wenbo Mao
Trusted E- Services Laboratory
HP Laboratories Bristol

HPL-2001-303
November 27th , 2001*

E-mail: Steven.Galbraith@rhul.ac.uk, wm@hplb.hpl.hp.com

We introduce a new security property for undeniable and confirmer signatures in the multi-user setting, namely, anonymity. We show that many existing systems do not have this property. We modify the original undeniable signature scheme of Chaum and van Antwerpen and the RSA-based undeniable / confirmer signature scheme of Gennaro, Krawczyk and Rabin so that they achieve anonymity. We also provide a new, more efficient, denial protocol for the undeniable signature scheme of Chaum and van Antwerpen.

* Internal Accession Date Only Approved for External Publication

[1] Mathematics Department, Royal Holloway University of London, Egham, Surrey TW20 0EX, UK

Anonymity and Denial of Undeniable and Confirmer Signatures

Steven D. Galbraith*[1] and Wenbo Mao[2]

[1] Mathematics Department, Royal Holloway University of London,
Egham, Surrey TW20 0EX, UK.
Steven.Galbraith@rhul.ac.uk
[2] Mathematics, Cryptography and Security Group
Hewlett-Packard Laboratories, Bristol
Filton Road, Stoke Gifford, Bristol BS34 8QZ, UK.
wm@hplb.hpl.hp.com

Abstract. We introduce a new security property for undeniable and confirmer signatures in the multi-user setting, namely, anonymity. We show that many existing systems do not have this property. We modify the original undeniable signature scheme of Chaum and van Antwerpen and the RSA-based undeniable/confirmer signature scheme of Gennaro, Krawczyk and Rabin so that they achieve anonymity. We also provide a new, more efficient, denial protocol for the undeniable signature scheme of Chaum and van Antwerpen.

1 Introduction

Undeniable signatures are public key digital signatures which cannot be verified without interacting the the signer. Confirmer signatures are undeniable signatures where signatures may be verified by interacting with an entity (the confirmer) designated by the signer. Implicit in this notion is the principle that information about the signature cannot be obtained without some interaction. The security of undeniable and confirmer signatures has been considered in the single user case under the notion of 'invisibility' [5], which is essentially the inability to determine whether a given message-signature pair is valid for a given user. In [5] invisibility is defined in terms of being able to be simulated. In [2] this notion is phrased in terms of distinguishing whether a signature s corresponds to a message m_0 or m_1.

In this paper we suggest that in the multi-user setting it is also important to consider the notion of 'anonymity'. Informally, this security property is as follows. Imagine a system with n users and suppose we are given a valid message-signature pair and are asked to determine which user generated the signature. By running signature confirmation or denial protocols with a given user (or their designated confirmer) we can determine whether or not they generated the signature. An undeniable or confirmer signature scheme has the anonymity property if it is infeasible to determine whether a user is or is not the signer of the message without interacting with that user or with the $n-1$ other users. A more precise definition of anonymity is given in Definition 1 where the problem is distilled down to the case of two users.

For example, one application of undeniable signatures is for open bid auctions. Suppose bidders submit public bids together with their undeniable signature on the bid. On completion of the auction the winning bidder can prove to the auctioneer that they signed the winning bid. Similarly, other bidders can prove that they are not the winners. In order to preserve anonymity of the bidders it is essential that it be computationally infeasible to guess the identity of a bidder from the published signature.

As this example shows, anonymity is an important security property for undeniable and confirmer signatures. The analysis of this property has been overlooked in the existing literature. The main purpose of this paper is to give a precise definition for anonymity in this setting and to provide practical and efficient schemes with this security property.

* This author thanks Hewlett-Packard Laboratories, Bristol for support.

Many existing schemes do not provide anonymity. For instance, as we show below, the undeniable signature scheme of Chaum and van Antwerpen does not provide anonymity, but may be easily modified (see Section 4) so that it has this property. The RSA-based schemes of Gennaro et al [8] and Galbraith et al [7] also do not provide anonymity. These schemes may be modified so that they provide anonymity, but this process is considerably more subtle than the finite field case.

For confirmer signature schemes we should stress that we do not study the problem of whether the signature reveals who the designated confirmer is (though our solutions do have this property).

1.1 The Chaum and van Antwerpen undeniable signature scheme

As a taster for the rest of the paper we now recall the scheme of [3] and show that a naive implementation of it in the multi-user setting does not provide anonymous signatures. Note that, in [3], the anonymity aspect was not explicitly discussed.

The undeniable signature scheme of [3, 4] is as follows. Let $g \in \mathbb{F}_p^*$. Each user has a secret key x and a public key $h = g^x$. The undeniable signature on message m is $s = H(m)^x$ where $H(m)$ is a one-way padding scheme involving a hash function. To confirm the signature one must interact with the signer and perform an interactive proof of equality of discrete logarithms on the tuple $(g, h, H(m), s)$. The signer must also have the ability to deny invalid signatures by performing a 'proof of inequality' protocol.

Assume that all users work in the same finite field (we discuss in Section 8 the harder case where users do not share the same group). In [3, 4] it was specified that g have prime order, but the values taken by m were not explicitly specified.

The problem of determining if a signature was generated using a certain user's public key is essentially the decision Diffie-Hellman problem. As is well known, this problem is not necessarily hard in groups whose order has small prime factors. If users allow $H(m)$ to be such that $(\frac{H(m)}{p}) = -1$ then, by considering the Legendre symbol of a known valid signature s, one can determine whether the secret key x of a given user is even or odd. One can construct a database of parity information about the various users in a system by considering known confirmed signatures. Then, for some other message-signature pair (m, s) such that $(\frac{H(m)}{p}) = -1$ one can reduce the number of candidate signers of the message by considering the value $(\frac{s}{p})$. Users whose secret key has the opposite parity can be eliminated from the list of possible signers without having to execute denial protocols with them. This shows that one cannot have anonymity without insisting that messages be such that $(\frac{H(m)}{p}) = +1$.

1.2 Plan of the paper

In Section 2 we mention some other undeniable and confirmer schemes and show why some of them do not provide anonymity. In Section 3 we give a precise definition for anonymity.

In Section 4 we give a modification of the undeniable signature scheme of Chaum and van Antwerpen and introduce a new denial protocol which is more efficient than any previous schemes in the literature. We prove that our new scheme has the anonymity property under the assumption that the Decision-Diffie-Hellman problem is hard.

We then discuss anonymity for some RSA-based undeniable and confirmer signature schemes. Our main result is Theorem 3 which shows that a modified version of the RSA-based schemes of Gennaro, Krawczyk and Rabin [8] and Galbraith, Mao and Paterson [7] have the anonymity property. The proof of Theorem 3 contains an innovative proof technique which will be of interest to other problems in provable security.

In Section 8 the scheme of Chaum and van Antwerpen is generalised to the case where users may use different finite fields. We show that anonymity may be achieved in this setting. In Section 9 we discuss the anonymity of several other schemes in the literature.

Finally, in Section 10, we discuss anonymity in the extremely general situation where participants may use completely different undeniable and confirmer signature schemes. We argue that anonymity can be obtained even in this setting, as long as certain conditions on the schemes are satisfied.

2 Previous schemes

We will discuss the following schemes.

1. The original undeniable signatures of Chaum and van Antwerpen [3,4].
2. The RSA-based undeniable signature of Gennaro, Krawczyk and Rabin [8] (which also has a confirmer version) and its extension by Galbraith, Mao and Paterson [7].
3. The original Chaum confirmer signature scheme [6].
4. The schemes of Michels and Stadler [11].
5. The scheme of Camenisch and Michels [2].

We now briefly summarise these schemes. The original Chaum and van Antwerpen scheme has already been mentioned in the introduction. The RSA-based schemes of [8,7] use the standard RSA signature $s = H(m)^d \pmod{N}$ but the difference is that the verification exponent e is not public but is instead known only to the signer (and confirmer).

The Chaum confirmer scheme [6] uses a discrete logarithm system with generator g and uses RSA signatures. The public key of the signer is (N, e). The confirmer has public key $h = g^x$. The signer chooses a random r, computes $a = g^r, b = h^r$ and $\alpha = (H(a, b) \oplus F(m))^d \pmod{N}$ (where F is a hash function and H is an invertible mixing function). The signature is (a, b, α). Both the signer and the confirmer are able to prove to another user that the signature is valid (see [6] for details).

Michels and Stadler give two solutions, both using the tool of 'confirmer commitments'. The first uses 3-move zero knowledge proofs for signature while the second uses existentially forgeable signature schemes.

Camenisch and Michels give a general construction built from a signature scheme and an encryption scheme. Let the confirmer have a public key K_C for the encryption scheme and a signer have a public key K_S for the signature scheme. To sign a message m the signer computes $s = \mathrm{Sign}(m)$ and $e = \mathrm{Enc}(s, K_C)$ and publishes e as their signature. The confirmer can decrypt e to obtain s and thus determine the validity of the signature using K_S. The confirmer is able to prove the validity to other entities using some zero-knowledge proof, which in general requires binary challenges and is very inefficient. Our solutions are much more efficient than the methods of Camenisch and Michels.

We will now mention the anonymity properties of these schemes. As shown above, the original undeniable signature scheme has an attack on anonymity if message digests $H(m)$ are permitted to have maximal order. We give a modified version of the scheme and prove that it has the anonymity property in Sections 4 and 5. Similarly, the RSA-based schemes of [8,7] have an attack on anonymity which is presented in Section 6. The schemes of [8,7] are repaired and their security proven in Sections 6 and 7. In Section 9 we show that Chaum's confirmer scheme [6] and one of the schemes of Michels and Stadler [11] (the one using RSA signatures) have attacks on anonymity of the same form as the attack on the other RSA-based schemes. In Section 9 we also argue that the Camenisch-Michels scheme [2] does have the anonymity property.

We must note that there is some disparity in the literature about the definition of confirmer signature schemes. The bone of contention is whether the original signer has the ability to confirm and/or deny signatures. Camenisch and Michels [2] claim that it is undesirable for signers to be able to verify or deny their signatures. We have a contrary opinion, that it is important for signers to be able to confirm and/or deny signatures. In particular, it seems to us to be an important personal right to be able to 'clear one's name' by denying signatures that are not genuine. The schemes of [2,11,6] do not allow users to deny signatures, whereas the schemes of [3,4,8,7] do allow this. In any case, these distinctions are not an obstacle to a discussion of the anonymity properties of the schemes.

3 Anonymity

The distinguishing property of undeniable and confirmer signatures [3,4,6] is that a signature cannot be verified without access to the signer or confirmer. This opens the possibility of having a system whereby signatures are anonymous, in the sense that no information about who generated a valid message-signature pair can be obtained apart from the one bit obtained by each execution of a confirm or denial protocol.

We will give a rigorous definition for anonymity. Constructing such a definition is rather complicated as there is the issue of whether signatures are valid or invalid for any user.

The first step is to distill the problem down to the case of just two users (a scheme with the anonymity property for two users can easily be shown to be secure in the case of n users).

If a signature is known to be valid for some user then the identity of the signer can be obtained by executing a signature confirmation protocol with that user, or by executing a signature denial protocol with the other user.

If a signature is not known to be valid for some user then one might expect that the problem be even harder, since executing denials with all but one user does not give any information about whether the user is the signer or not. But there are many reasons why a signature might not be valid for any users, and some of these might conceivably be easily determined by an adversary. For our definition we specifically exclude this case by imposing the condition that the signature be valid for one of the users.

Definition 1. (Anonymity) *Let* (Gen,Sign,Confirm,Deny) *be an undeniable or confirmer signature scheme. An adversary* $\mathbf{D}$ *is said to be a* **distinguisher** *under a chosen message attack if it behaves as follows.*

Let $(pk_0, sk_0) \leftarrow \mathrm{Gen}(1^k)$ *and* $(pk_1, sk_1) \leftarrow \mathrm{Gen}(1^k)$ *be two key pairs. The input to* $\mathbf{D}$ *is the pair* (pk_0, pk_1)*. The distinguisher* $\mathbf{D}$ *is permitted to interact with the hash function oracle(s), to obtain signatures on messages and to run signature verification and denial protocols (with the signer or a confirmer as appropriate) with respect to both of these public keys. At some point* $\mathbf{D}$ *constructs a message* m *and requests a challenge signature* $s \leftarrow \mathrm{Sign}_{sk_b}(m)$ *where the bit* $b \in \{0,1\}$ *is hidden from* $\mathbf{D}$*. The interaction with the cryptosystem continues with the exception that verification and denial protocols cannot be executed on the challenge message-signature pair* (m, s)*. The output of* $\mathbf{D}$ *is a guess* b' *for the hidden bit* b*.*

A distinguisher $\mathbf{D}$ *with output* b' *is said to be successful with advantage* $\epsilon(k)$*, if with probability at least* $1/2 + \epsilon(k)$*, we have* $b' = b$*.*

An undeniable or confirmer signature scheme has the anonymity property if there is no distinguisher which runs in polynomial time and has a non-negligible advantage.

It is interesting to contemplate the relationship between the definition of anonymity given above and the definitions of invisibility given in [5, 2]. We believe that, as stated, there is no relationship between these notions. Understanding the relationship between anonymity and invisibility is an interesting topic for future research.

4 Revised undeniable signature scheme

We now consider how to ensure that the undeniable signature scheme of Chaum and van Antwerpen has the anonymity property in the case where all users share the same finite field (see Section 8 for the case of different finite fields).

The first step is obviously to ensure that the attack described in the introduction cannot be applied. This attack relied on the presence of elements of order two, but more general versions can be applied using elements of small order, since the factorisation of $p-1$ is assumed to be public. Clearly, the attack generalises from prime fields $\mathbb{F}_p$ to more general finite fields $\mathbb{F}_q$ where $q = p^n$.

There are two ways to proceed, one is to blind the signature using elements of small order (this is the strategy used in later sections of the paper). Since the factorisation of $q-1$ is known (and is shared by all users) it is equivalent to work in a subgroup of large prime order l of the finite field $\mathbb{F}_q^*$. In this case all elements will have Legendre symbol $+1$. It is therefore necessary to modify the padding scheme $H(m)$ so that it takes values in this subgroup. This is easily done by raising the value of $H(m)$ to the cofactor $(q-1)/l$.

As we consider strong active attack models it is necessary that the value for $H(m)$ be randomised. In this paper we assume that a randomised padding scheme such as that given in [1] is used. The padding scheme of [1] enables us to obtain our security reductions in the random oracle model.

To obtain the security result it is necessary that executions of the confirm and deny protocols can be simulated. This is not possible with interactive proofs so we must use non-interactive proofs. To maintain the security of the system (i.e., that proofs cannot be transferred to other users) it is necessary to use designated-verifier proofs [9]. Such proofs can be simulated in the random oracle model. For further details see Jakobsson et al [9, 10].

For the sake of completeness we give the signature confirmation protocol of [9] for the challenge $(g, h, H(m), s)$ where $g, h, H(m)$ and s all have order l and where $h = g^x$. We also use the public key $y \in \langle g \rangle$ of the designated verifier (this is for the trapdoor commitment scheme).

1. Prover chooses random w, r, t and computes $c = g^w y^r, G = g^t, M = H(m)^t$, $b = H'(c, G, M)$ and $d = t + x(b + w) \pmod{l}$ where H' is some cryptographically strong hash function with full domain output onto $\mathbb{Z}/l\mathbb{Z}$.
2. Prover sends (w, r, G, M, d) to the verifier.
3. Verfier computes $c = g^w y^r$ and $b = H'(c, G, M)$ and checks that $Gh^{b+w} = g^d, Ms^{b+w} = H(m)^d$.

We refer to [9] for the further discussion of this protocol. We note that it is implicit in the above that the verifier knows the value $H(m)$. Hence this value must be transmitted if the verifier cannot deterministically calculate it.

We now introduce a new denial protocol for this scheme which is of interest as it is more efficient than any of the previous methods proposed.

The signature denial protocol runs as follows:

1. Prover chooses a random $1 \leq v < l$ and publishes $t_1 = g^v$, $t_2 = H(m)^v$ and $t_3 = v/x \pmod{l}$.
2. Prover sends the designated-verifier, non-interactive proof of signature confirmation to the verifier with respect to $(g, t_1, H(m), t_2)$ (i.e., shows that it is a valid Diffie-Hellman tuple).
3. Verifier checks the confirmation proof, checks that $t_1 = h^{t_3}$ and checks whether $t_2 \overset{?}{=} s^{t_3}$.

What this protocol is doing is blinding the challenge $(g, h, H(m), s)$ and then publishing the signature t_2 for $H(m)$ with respect to the 'new' public key t_1. Note that giving a valid signature for the random public key t_1 is not a leakage of information since any adversary can obtain such a signature themselves by choosing a random v.

Theorem 1. *The denial protocol given above is a non-interactive, designated-verifier proof of (in)validity of signatures. It is sound if the discrete logarithm problem is hard and it is zero knowledge in the random oracle model.*

Proof. The completeness of the protocol (i.e., that an honest verifier accepts the proof) is clear.

The protocol is designated verifier since knowledge of the secret key z such that $y = g^z$ allows one to create transcripts easily.

To show that the protocol is zero knowledge in the random oracle model we show how to simulate transcripts. First choose t_3 at random, set $t_1 = h^{t_3}$ and set t_2 to be (not) equal to s^{t_3} depending on whether one wants to confirm or deny. Then choose d, b, w randomly, set $G = g^d / t_1^{b+w}$, $M = H(m)^d / t_2^{b+w}$ and choose r randomly. Now, define the value of the random oracle H' at $(g^w y^r, G, M)$ to be b.

Soundness follows easily from the soundness of the confirmation protocol. □

The cost of the protocol is two exponentiations more (for both prover and verifier) than a run of the confirm protocol. This is the most efficient denial protocol in the literature for the undeniable signature scheme of Chaum and van Antwerpen.

We note that the revised scheme has all the other desirable security properties of an undeniable signature scheme (see Chaum et al [3, 4], Camenisch and Michels [2] and Okamoto and Pointcheval [12]).

5 Anonymity of revised undeniable signature scheme

The modified Chaum undeniable signature scheme clearly avoids the attack mentioned earlier since all elements have Legendre symbol +1. We now want to prove the anonymity of this scheme under the very strong adaptive attack model and the assumption that the decision Diffie-Hellman problem (DDH) in the subgroup of $\mathbb{F}_q^*$ of large prime order l is hard.

Theorem 2. *Suppose two players use the modified Chaum undeniable signature scheme above in the same subgroup of order l in $\mathbb{F}_q^*$. Suppose the Decision Diffie-Hellman problem in this subgroup is hard. Then, in the random oracle model, the signature scheme has the anonymity property under an adaptive chosen message attack.*

Proof. Suppose that an adversary $\mathbf{D}$ to the undeniable signature scheme exists. We will transform it into a DDH algorithm. Let the input DDH problem be (g_1, g_2, g_3, g_4).

We first set up two public keys with generator g_1. The first has $h_1 = g_2$ and the second has $h_2 = g_2^a$ for some randomly chosen integer a.

We now run the adversary $\mathbf{D}$ on this public key pair. The adversary will expect to consult hash function and signing oracles and will also expect to engage in runs of the confirmation and denial protocols. At some point $\mathbf{D}$ will produce a message m and request a challenge undeniable signature from one of the two public keys. We must simulate all these operations.

Hash query: When the adversary $\mathbf{D}$ makes a hash query on m we check whether $H(m)$ has already been defined. If not, then random integers x and y are chosen and, in a standard way (see [1]) $H(m)$ is defined to be $g_1^x g_3^y$.

Signing query: When $\mathbf{D}$ makes a signing query with a message m we first ensure that $H(m)$ is defined and obtain the matching values of x and y. If the query is with respect to public key 1 then output $s = g_2^x g_4^y$. If the query is with respect to public key 2 then output $s = (g_2^x g_4^y)^a$. In the case of the challenge message we choose a bit b at random and sign with respect to that public key value using the above method.

Confirm/Deny: When the adversary wants to engage in a signature confirmation or denial on a pair (m, s) with respect to public key i we first have to determine whether the signature is valid or not.

Within the simulation we can do this by consulting the state information and seeing what value (if any) has been specified for $H(m)$ and whether or not the value s is the valid simulated signature for that hash value. If $H(m)$ is not specified then choose random x and y and define $H(m) = g_1^x g_2^y$ and declare the signature to be invalid.

Once the validity (within the simulation) of the signature has been determined we know whether to respond positively or negatively to the execution of the confirmation or denial protocol. Since the zero knowledge proofs are perfectly simulatable in the random oracle model we can easily construct a proof which gives a suitable response.

Finally, the adversary will output its guess b' to the value of the bit b. If $b = b'$ then output the result 'true' for the validity of the Diffie-Hellman tuple, and if $b \neq b'$ then output 'false'.

When the input is a valid Diffie-Hellman tuple then the simulation is identical to a genuine attack on the cryptosystem. Hence the advantage for the Decision-Diffie-Hellman algorithm is exactly the same as the advantage of $\mathbf{D}$.

When the input is not a valid Diffie-Hellman tuple then transcript of values for $H(m)$ and s is indistinguishable from a uniform distribution (since for all $u, v \in \langle g_1 \rangle$ there is some x, y such that $g_1^x g_3^y = u$ and $g_2^x g_4^y = v$). Hence the transcript is independent of the hidden bit b. This means that the adversary $\mathbf{D}$ has no chance of correctly guessing the signer and the probability that $b = b'$ is $1/2$.

The above argument includes the case where $\mathbf{D}$ detects that the simulation is invalid. Regardless of what strategy is used by $\mathbf{D}$ in this case the probability that $b = b'$ is $1/2$.

Finally, it is clear that the possibility of guessing a correct s from m or $H(m)$ is negligible so one can ignore the possibility that the some executions of the confirm or denial protocols might actually fail when, under the simulation, they should succeed. □

In the full version we will make the complexity of this reduction more precise, though it is obvious that the computational complexity and advantage are basically the same as that for **D**.

6 Undeniable signatures based on RSA

Gennaro, Krawczyk and Rabin [8] described an undeniable/confirmer signature scheme based on RSA. In their case the signature for a message m is the number s such that $s \equiv H(m)^d \pmod{N}$. The verification exponent is fixed by publishing the value $h = g^d \pmod{N}$. This scheme was generalised by Galbraith, Mao and Paterson [7], who also gave a more efficient denial protocol.

Since d is odd it follows that the Jacobi symbols $(\frac{s}{N})$ and $(\frac{H(m)}{N})$ are equal. Hence, given a pair $(H(m), s)$ and a set of users' public keys $\{N_i\}$ one can eliminate some candidate signers by checking if $(\frac{s}{N_i}) \neq (\frac{H(m)}{N_i})$. This leads to an increased probability of guessing who the actual signer is, which means that the scheme does not have the anonymity property. Similarly, the scheme of Galbraith, Mao and Paterson [7] does not have the anonymity property.

6.1 Preventing the Jacobi symbols attack

The question therefore arises as to whether the schemes of [8] and [7] can be modified to give anonymity. Note that it is tempting to enforce that $H(m)$ be a quadratic residue in $\mathbb{Z}_N^*$ (and so s would also be a quadratic residue in $\mathbb{Z}_N^*$) but this does not provide a solution since the anonymity could still be broken by computing $(\frac{s}{N})$ and eliminating those N for which the value is -1.

A better solution is to define signatures by

$$s = \xi H(m)^d \pmod{N}$$

where ξ is a randomly chosen square-root of 1 in $\mathbb{Z}_N^*$. This choice of s means that there is no longer necessarily any relationship between $(\frac{H(m)}{N})$ and $(\frac{s}{N})$. The verification operation is to check that $H(m)^2 \equiv s^{2e} \pmod{N}$ where the verification exponent e is only known to the signer and confirmer.

The first observation is that this is dangerous, since two signatures s and s' on the same message leak a square-root of unity $s/s' = \xi/\xi'$. However, this scenario never arises since, to have security in an adaptive attack model, the value $H(m)$ must be randomised.

We give further details on how $H(m)$ is constructed, following [1]. We want $H(m)$ to be a k-bit string where k is longer than the modulus length. Choose values k_0 and k_1 and choose hash functions h, g_1, g_2 with output lengths k_1, k_0 and $k - k_0 - k_1$ respectively. To compute $H(m)$ we choose a random k_0-bit string r, compute $w = h(m, r)$, $r^* = r \oplus g_1(w)$ and $\gamma = g_2(w)$. The output value for $H(m)$ is $w \| r^* \| \gamma$. Thus, two signatures on the same message m will yield completely different values for $H(m)$ when the values for r are different.

One consequence of using a randomised padding scheme in the context of undeniable signatures is that it is necessary to transmit the value $H(m)$ as part of the signature. This is because, unlike with standard RSA signatures, the value $H(m)$ is not recovered by the verifier as part of the signature verification process.

6.2 Ensuring that the signature length does not reveal the signer

There is a further attack on anonymity of RSA-based schemes which arises since all users must have different moduli N. If the moduli have different sizes then some values for s might be too large to have arisen from the signature process of some users. This reduces the number of possibilities for the signer.for proofs in the random oracle model A solution to this problem is for users to enlarge any values $s \pmod{N}$ to a fixed bitlength by adding a suitable multiple of N and transmitting $s' = s + tN$. This padding removes any information about the size of N and does not interfere with the reduction of the value modulo N. Let k be a bitlength such that there is a multiple TN close to 2^k (this is easily achieved if k is significantly larger than the bitlength of N). Then for signatures $s \in \mathbb{Z}_N^*$ and random $0 \leq t < T$ it follows that $s + tN$ is indistinguishable from a random k-bit string. We assume that this technique is adopted for the scheme below.

6.3 Summary of the modified scheme

We also make two modifications to the RSA-based undeniable/confirmer signature schemes of [8,7]. First we impose that condition that the modulus is a Blum integer (i.e., product of two primes congruent to 3 modulo 4). Generalising to non-Blum integers is straightforward. Also, as in the Chaum and van Antwerpen case above, we insist that these schemes are provided with non-interactive, designated-verifier proofs which are simulatable in the random oracle model.

In summary, the scheme is as follows.

System parameters: A fixed bitlength k such that all users' moduli are at most k bits. A randomised padding scheme as in [1] with output bitlength k for all users. A soundness bound B as in [7].

Key generation: A signer chooses two primes $p \equiv q \equiv 3 \pmod 4$ such that all prime factors of $(p-1)/2$ and $(q-1)/2$ are greater than B. The signer sets $N = pq$ and chooses $e, d \in \mathbb{Z}$ such that $ed \equiv 1 \pmod{\varphi(N)}$. The signer chooses $g \in \mathbb{Z}_N^*$ (or possibly many g_i) and sets $h = g^d \pmod N$ (or $h_i = g_i^d$). In the following we assume that there is a single pair (g, h), the modifications to the general case are trivial. The signer registers with the certificate authority as in [7] with public key (N, g, h). The signer sends e to the designated confirmer (if there is one).

Signing: To sign a message m the signer constructs the randomised padding value $H(m)$ (e.g., chooses r at random and computes $w = h(m,r), r^* = r \oplus g_1(w), \gamma = g_2(w)$ and $H(m) = w \| r^* \| \gamma)$. The signer computes $s = \xi H(m)^d \pmod N$ where $\xi \in \mathbb{Z}_N^*$ is a random element of order 2. The signer enlarges s to a bitstring s' of length k by adding a suitable random multiple of N. The signature on m is the pair $(H(m), s')$ of k-bit strings.

Confirm/Deny: To confirm or deny a signature the confirmer executes non-interactive, designated verifier versions of the proofs in [7] which prove the relationships $g \equiv h^e \pmod N$ and $s^{2e} \overset{?}{\equiv} H(m)^2 \pmod N$.

We note that this scheme preserves the strong security properties of the schemes in [8,7].

7 Anonymity of revised RSA-based undeniable and confirmer signatures

We will show that the anonymity of the system depends on the following computational problem:

Special Composite Decision Diffie-Hellman Problem (SCDDH): Let N be a Blum integer (i.e., $N = pq$ with $p \equiv q \equiv 3 \pmod 4$ both prime). Let $g, h, u, v \in \mathbb{Z}_N^*$ be such that g and u generate $\mathbb{Z}_N^*$, (and so $(\frac{g}{N}) = (\frac{u}{N}) = -1$), $(\frac{v}{N}) = +1$ and $h = g^d \pmod N$ for some (unknown) integer d coprime to $\varphi(N)$. Determine whether or not $v \equiv \xi u^d \pmod N$ for some element $\xi \in \mathbb{Z}_N^*$ of order 2.

We claim that this problem is computationally intractable when N is a product of two large primes and when g, u, v and d are chosen uniformly at random subject to the conditions.

Theorem 3. *In the random oracle model then the RSA-based undeniable/confirmer signature scheme above has the anonymity property if the special composite decision Diffie-Hellman problem is hard.*

Proof. Suppose we have an adversary $\mathbf{D}$ to the scheme. Let (N_1, g_1, h_1, u, v) be a SCDDH challenge problem. We will transform $\mathbf{D}$ into an algorithm to solve this SCDDH instance.

We let (N_1, g_1, h_1) be the first public key. In the general case where several pairs (g_i, h_i) are required then we construct them to be of the form $g_i = g_1^x u^y$ and $h_i = h_1^x v^y$. Then set up a second public key (N_2, g_2, h_2) using the key generation process for the scheme (in particular, we know the secret key d_2 for this public key).

We then execute the distinguisher $\mathbf{D}$ on these two public keys after randomly switching the indices 1 and 2 (in this proof index 1 will always refer to the public key coming from the SCDDH challenge, but this may be the first or second input to **D**). The distinguisher expects to perform hash queries, to obtain signatures on messages of its choice, and to run confirm and denial protocols. We now show how these will be simulated.

Hash query: A hash query could be with respect to any of the random oracles h, g_1 or g_2. We first analyse how to respond to a query of $h(m, r)$ where m is a message and r is a random k_0-bit string.

If the value $h(m, r)$ has not been queried before we choose x and y at random between 1 and N_1 and compute a k-bit string which reduces modulo N_1 to $g_1^x u^y$. All values are stored as state information. This string is parsed as $w \| r^* \| \gamma$ and $h(m, r)$ is defined to be w. Similarly, $g_1(w)$ is defined to be r^* and $g_2(w)$ is defined to be γ. If there are any conflicts with previous definitions of the random oracles then the simulation halts (or retry with a different x, y choice); this happens with negligible probability if k, k_0 and k_1 are large enough.

A query on g_1 or g_2 with input value w can be answered as above when w is the output of a previous query on $h(m, r)$, and can be answered with a random bitstring otherwise.

Since g and u generate $\mathbb{Z}_{N_1}^*$ it follows that the output of all the random oracles is indistinguishable from uniform random.

Signature query: To sign m we first check whether $H(m)$ has been queried or not (if not, perform the above process) and obtain the corresponding values x and y from the state information. If we must sign with respect to public key 1 then output $s = h_1^x v^y \pmod{N_1}$. If we must sign with respect to public key 2 then output $s = \xi H(m)^{d_2} \pmod{N_2}$ which is the usual signing process.

For the challenge signature we using the above signing process using public key 1. This determines the hidden bit b.

Confirm/Deny: It is first necessary to decide whether the signature should be considered valid or not within the simulation. If the request is with respect to public key 2 then this is performed in the usual way. If the request is with respect to public key 1 then it is necessary to consider the hash value $H(m) = g_1^x u^y$ and check if s takes the value $h_1^x v^y$ or not.

Once we have determined whether to respond positively or negatively then an appropriate proof can be simulated in the random oracle model.

The distinguisher will eventually output a guess b' for the hidden bit b.

When the SCDDH problem is valid then the simulation is identical to a real attack on the system, and so **D** should ouput $b' = b$ with the same advantage as the advantage of the SCDDH algorithm

When the SCDDH problem is invalid then we cannot argue that the simulation is indistinguishable from a genuine run of the attack. Therefore, we cannot predict how **D** will behave in this situation.

To handle this situation we introduce a new technique for security proofs. The crucial observation is that, although we do not know how **D** behaves in this situation, we do know that **D** is an algorithm whose behaviour must be consistent across different executions of the game. Hence, we can experiment with **D** to determine how it behaves when the inputs are of a certain form. These experiments with **D** allow us to be able to predict its behaviour.

For the current application it is sufficient simply to repeat the entire procedure above, except that the challenge signature is now constructed to be of the form $s = h_1^{x'} v^{y'} \pmod{N_1}$ where x' and y' are chosen at random independently of the value $H(m) = g_1^x u^y \pmod{N_1}$.

We now consider the difference between the two games played with **D**. When the input is a valid SCDDH tuple then the first game is identical to a real attack while the second game does not give **D** any information about who is the signer of the challenge (it is an invalid signature) and so the advantage of **D** will be 1/2. When the input is an invalid SCDDH tuple then both games are indistinguishable and so the outputs of **D** should be indistinguishable.

Write b_1 and b_2 for the hidden bits and write b_1' and b_2' for the outputs by **D** from the first and second games. If $b_1 = b_1'$ and $b_2 \neq b_2'$ then answer 'true' for the SCDDH question, otherwise output 'false'. When the input is a valid SCDDH tuple then the output of the simulator will be correct with some advantage. When the input is invalid then **D** responds with consistently reliable guesses for b'_i and so the output will be 'false' with probability at least $1/2$. Since $\mathbf{D}$ is supposed to have a non-negligible advantage it follows that we have a non-negligible advantage against the SCDDH problem, after roughly double the computation time. □

In the full version we will make the complexity of this reduction more precise, but it is obvious that the computational complexity is roughly double and that the advantage is roughly halved in the worst case.

8 Generalised Chaum and van Antwerpen scheme

In the previous sections we have shown how RSA-based schemes can provide the anonymity property even though each user is working in a different group. This opens the possibility that the scheme of Chaum and van Antwerpen could also be developed in a situation where users do not share the same finite field. In this section we show how to achieve this without any loss of security.

First we show that the scheme of Section 4 does not have the anonymity property if users do not all use the same finite field. In that setting the value s always has prime order l in $\mathbb{F}_q^*$. If two users have different fields $\mathbb{F}_{q_1}^*$ and $\mathbb{F}_{q_2}^*$ with corresponding primes l_1, l_2 then it is easy to determine whether a bitstring s corresponds to an element of order l_i in $\mathbb{F}_{q_i}^*$ or not, and so the anonymity of the scheme can be broken.

We revise the scheme again as follows. Fix a security parameter k for all users of the system. Let each user choose a prime (or prime power) q such that q is less than k bits long. Write $q - 1 = nl$ where l is a (large) prime and where n is some cofactor. Let $g \in \mathbb{F}_q^*$ have order l and let $h = g^x$. The public key for a user is (q, n, l, g, h) and the secret key is x. The signature on $H(m)$ is

$$s = \xi H(m)^x$$

where $\xi \in \mathbb{F}_q^*$ is an element of order dividing n. The signature confirmation and denial protocols are as before, testing the condition that $(g, h, H(m)^n, s^n)$ is a valid Diffie-Hellman tuple of elements of order l in $\mathbb{F}_q^*$.

The key point for security is that, since the output size of $H(m)$ is larger than all choices for finite fields q, any element $s \in \mathbb{F}_q^*$ can arise as some signature on some message with some blinding factor ξ.

Finally, it is necessary to extend the signature s to a bitstring of a fixed length k so that its length does not reveal the signer. When q is a prime then this is done by adding a suitable multiple of q. When q is a prime power then natural generalisations of this approach may be used, depending on the representation used for finite field elements.

We now show that this revised scheme has the anonymity property. The methods used to prove Theorem 3 can be easily adapted to prove the following result (in fact, the proof is easier than the proof of Theorem 3, since the simulator can construct elements ξ itself when the values n and l are known).

Theorem 4. *In the random oracle model then the generalised undeniable signature scheme above has the anonymity property if the decision Diffie-Hellman problem in a large prime order subgroup of $\mathbb{F}_q^*$ is hard.*

9 Other schemes

It is straightforward to show that the confirmer signature scheme due to Camenisch and Michels [2] has the anonymity property if the underlying encryption scheme is semantically secure under an adaptive chosen ciphertext attack (the method of proof is identical to the proof of invisibility in Theorem 1 of [2]).

In the confirmer signature scheme of Chaum [6] the signature includes a number α such that $\alpha^e \equiv m \pmod{N}$ where m is known (it is a function of the signature components a and b) and where (N, e) is the public key for a user. The Jacobi symbols attack given above applies in this situation too (just compare $(\frac{\alpha}{N})$ with $(\frac{H(a,b)\oplus F(m)}{N})$) and so this scheme does not have the anonymity property.

The RSA-based scheme of Section 5.4 of Michels and Stadler [11] does not have the anonymity property. In this scheme the signature is a usual RSA signature of a known value m (which is $h_B(d) + b$ in the setting of [11], where all quantities are visible to an adversary). Hence the same attack as above using values of the Jacobi symbol can be applied.

We note that the protocol in [11] based on Schnorr signatures seems to have the anonymity property.

10 Anonymity between signatures of different schemes

The above results show that undeniable and confirmer signature schemes can provide anonymity even when users work in different groups. This can be extended further: One can have anonymity even when users are using different undeniable and confirmer signature schemes!

There are two primary requirements which must be satisfied for a secure undeniable or confirmer signature scheme to be anonymous in this setting. The first is that there should be a fixed size k for bitstrings for all users, so that all signatures are a string of k bits and all public keys should specify groups (or output spaces) where the elements are represented by strings of less than k bits. The second requirement is that the schemes should be constructed so that the valid signatures are indistinguishable from random k-bit strings (even when using knowledge of a user's public key).

It is possible to obtain a security result in this very general setting using the same techniques as used to prove Theorem 3 and Theorem 4. A more formal definition, and the details of the proof will be given in the full version of the paper.

References

1. M. Bellare and P. Rogaway, The exact security of digital signatures - how to sign with RSA and Rabin, Advances in Cryptology: Proceedings of EUROCRYPT 96 (U. Maurer, ed.), Lecture Notes in Computer Science 1070, Springer-Verlag 1996, pages 399-416.
2. J. Camenisch and M. Michels, Confirmer signature schemes secure against adaptive adversaries, Advances in Cryptology: Proceedings of EUROCRYPT 2000 (B. Preneel, ed.), Lecture Notes in Computer Science 1870, Springer-Verlag 2000, pages 243-258.
3. D. Chaum and H. van Antwerpen, Undeniable signatures, Advances in Cryptology: Proceedings of CRYPTO 89 (G. Brassard, ed.), Lecture Notes in Computer Science 435, Springer-Verlag 1990, pages 212-216.
4. D. Chaum, Zero-knowledge undeniable signatures, Advances in Cryptology: Proceedings of CRYPTO 90 (I.B. Damgaard, ed.) Lecture Notes in Computer Science 473, Springer-Verlag 1991, pages 458-464.
5. D. Chaum, E. van Heijst and B. Pfitzmann, Cryptographically strong undeniable signatures, unconditionally secure for the signer, in J. Feigenbaum (ed.), Crypto '91, Springer Lncs 576, (1992) 470–484.
6. D. Chaum, Designated confirmer signatures, Advances in Cryptology: Proceedings of EUROCRYPT 94 (A. de Santis, ed.), Lecture Notes in Computer Science 950, Springer-Verlag 1995, pages 86-91.
7. S. D. Galbraith, W. Mao, and K. G. Paterson, RSA-based undeniable signatures for general moduli, to appear in the proceedings of RSA 2002.
8. R. Gennaro, H. Krawczyk and T. Rabin, RSA-based undeniable signatures, Advances in Cryptology: Proceedings of CRYPTO 97 (W. Fumy ed.), Lecture Notes in Computer Science 1294, Springer-Verlag 1997, pages 132-149. Also in *Journal of Cryptology* (2000)13:397–416.
9. M. Jakobsson, K. Sako and R. Impagliazzo, Designated verifier proofs and their applications, in U. Maurer (ed.) EUROCRYPT '96, Springer LNCS 1070 (1996) 143–154.
10. M. Jakobsson, Efficient oblivious proofs of correct exponentiation, in B. Preneel (ed.), Communications and multimedia security, Kluwer (1999) 71–84.
11. M. Michels and M. Stadler, Generic constructions for secure and efficient confirmer signature schemes, in K. Nyberg (ed.) EUROCRYPT '98, Springer LNCS 1403 (1998) 406–421.
12. T. Okamoto and D. Pointcheval, The Gap-problems: a new class of problems for the security of cryptographic schemes, in K. Kim (ed.) PKC '2001, Springer LNCS ??? (2001)