Robust and Efficient Sharing of RSA Functions

Rosario Gennaro* Stanislav Jarecki† Hugo Krawczyk‡ Tal Rabin§

Abstract

We present two efficient protocols which implement robust threshold RSA signature schemes, where the power to sign is shared by N players such that any subset of more then T signers can collaborate to produce a valid RSA signature on any given message, but no subset of fewer than T corrupted players can forge a signature. Our protocols are robust in the sense that the correct signature is computed even if up to T players behave in arbitrarily malicious way during the signature protocol. This in particular includes the cases of players that refuse to participate or that generate incorrect partial signatures. Our protocols achieve fault tolerance T of $N/2$, which is optimal. Our protocols are also very efficient, as the computation performed by each player is comparable to the computation cost of a single RSA signature.

Robust threshold signature schemes have very important applications, since they provide increased security and availability for a signing server (e.g. a certification authority.) Solutions for the case of the RSA signature scheme are especially important because of its spread use. In addition, solutions to shared RSA signatures usually lead to efficient key escrow schemes for RSA.

The scheme is based on some interesting extensions that we devised for the Information Checking Protocol of T. Rabin and Ben-Or [Rab94, RB89], and the *undeniable signature* work of Chaum [CA90]. These extensions have desirable properties, and hence are of independent interest.

* Lab. of Computer Science, Massachusetts Institute of Technology, 545 Tech Square, Cambridge, MA 02139, USA. Email: rosario@theory.lcs.mit.edu.

† Lab. of Computer Science, Massachusetts Institute of Technology, 545 Tech Square, Cambridge, MA 02139, USA. Email: stasio@theory.lcs.mit.edu.

‡ IBM T.J. Watson Research Center, PO Box 704, Yorktown Heights, New York 10598, USA. Email: hugo@watson.ibm.com.

§ Research supported by an NSF Post-doctoral Fellowship Lab. of Computer Science, Massachusetts Institute of Technology, 545 Tech Square, Cambridge, MA 02139, USA. Email: talr@theory.lcs.mit.edu.

1 Introduction

The idea of distributed signature schemes is to depart from the single-signer approach in which one person is the sole holder of a secret key, and to allow a group of people to "hold" the key in such a manner that they can, as a group, produce signatures, yet no person on his own can generate a signature. The signature which is generated by the group is the same as if it were generated by a single signer.

We say that a distributed signature scheme is a (T, N)-*threshold signatures scheme*, if given a message m, and a group of N players, where each one holds a part of a secret key, any subset of T or more players can generate the signature for m. We say that the scheme is *secure/unforgeable* if no coalition of fewer than T players can produce a valid signature [GMR88], even after the system has produced many signatures for different messages.[1] Furthermore, a (T, N)-threshold signatures scheme is *robust/fault-tolerant* if it can correctly compute signatures, even in the presence of up to T arbitrarily malicious players.

Note that a simple reconstruction of the key at signing time would not satisfy our requirement, as it allows future signatures to be generated by a single player.

Robust threshold signature schemes have very important applications, and we shall exemplify this briefly. The secret key of the government's certification authority will be an attractive target for attacks. If the key is held in one site then once this site is broken into, the key is exposed. Yet, if the key is distributed among N sites, using a (T, N)-threshold signature scheme, then one needs to break into T sites in order to learn the key and forge signatures. In addition, once a site is broken into, it might exhibit arbitrary performance, yet the system should still be able to generate signatures. Hence, we increase the security and robustness of a system by distributing the secret key. Furthermore, if the key is held on a single site, then signatures cannot be generated if this site crashes. Note that the trivial solution of replication solves the availability problem, yet generates more sites that hold the full key. A threshold signature protocol can in particular tolerate up to T such crashes, thus increasing the availability of the signature operation, without decreasing its security.

Threshold signatures are part of the general approach known as *threshold cryptography* introduced through the works of Boyd [Boy86], Desmedt [Des94], and Desmedt and Frankel [DF90]. Solutions for the case of the RSA signature scheme are especially important because of its spread use (a de-facto standard). In addition, since in the RSA cryptosystem the signing algorithm coincides with the decryption algorithm, solutions to shared RSA signatures usually lead to shared RSA decryption procedures which have applications to key escrow (See [Mic92]). Desmedt and Frankel initiated the study of threshold-RSA [DF92], which was followed by [DDFY94]. In these papers, protocols for $(N/2, N)$-threshold RSA signatures were given. However, these schemes are not robust.

In this paper, we present an efficient solution for robust, $(N/2, N)$-threshold RSA signatures. The work of [DDFY94] is used as the basic threshold RSA scheme. We achieve the additional property of fault-tolerance by employing extensions which we developed of the Information Checking Protocol of T. Rabin and Ben-Or [Rab94, RB89], and the *undeniable signature* work of Chaum [CA90]. These extensions have desirable properties, and hence can be used in other applications as well.

In a recent and independent work, Frankel, Gemmel, and Yung [FGY96], have extended the notion of *result-checking* introduced by Blum [BK89], to the setting of *witness-based cryptographic checking*. Among the main motivations for that work is the generation of a robust fault-tolerant RSA signature scheme. While [FGY96] provides a more general theoretical framework, our techniques, specifically

[1] Unforgeability can be achieved by enforcing a stronger requirement on the protocol, which is that it be *zero-knowledge* with respect to the adversary [GMR89].

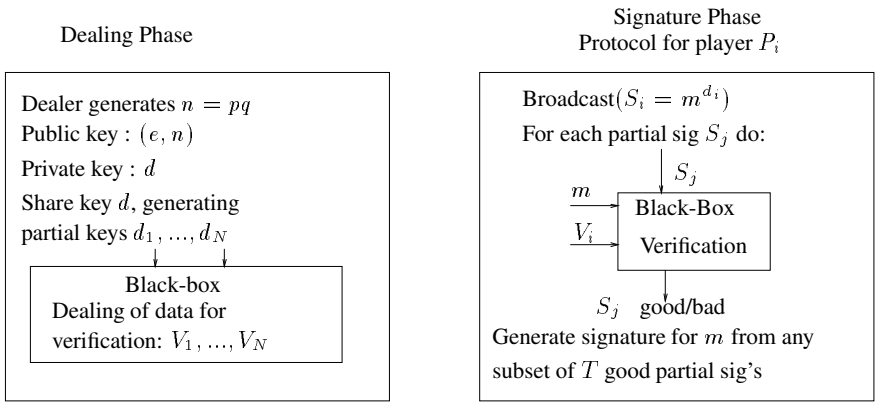
designed for RSA, result in much more efficient and practical solutions. In particular, our basic protocols involve just a small constant number of modular exponentiations while in [FGY96] a very large number, proportional to the security parameter, of such costly exponentiations is required.

2 Our results

The basic construction underlying the existing threshold RSA schemes can be roughly described as follows: given $n = pq$, the public RSA key is set to (n, e), where $(e, \phi(n)) = 1$ and the secret key d, is such that $ed \equiv 1 \bmod \phi(n)$, and the signature on a message m is $S = m^d \bmod n$. The distributed signature scheme has two phases. In the first one, called the Dealing Phase, a dealer shares the secret key d, among the N players, such that each player P_i has a "share" d_i of d. These shares are created in such a manner, that in the second phase, which is called the Signature Phase, when a message m is given, any subset of T partial signatures $S_i = m^{d_i} \bmod n$ suffice to generate the signature for m. Further details of the result by [DDFY94] in Section 3.1.

Each of the existing schemes *requires* that all partial signatures, S_i be correct, hence these schemes cannot tolerate faults. If we are to confront failures, we must be able to detect which of the partial signatures are improper. Once the incorrect partial signatures are sieved out, the computation can be carried out in the same manner as in the case where there are no faults.

In this paper we concentrate on providing solutions for the problem of detecting an improper partial signatures. To achieve this goal, additional data $V_1, ..., V_N$ is generated in the Dealing Phase, where V_i is known to P_i. This data will be used in the Signature Phase is order to authenticate partial signatures. The following is a schema of a fault-tolerant, threshold RSA signature scheme:



The protocols which we present in this paper are the "black-boxes" shown in the schema. In particular we prove the following theorem:

Theorem *There exists an efficient, robust* $(N/2, N)$*-threshold RSA signature scheme.*

Note, that the activation of the verification black-box can be avoided under proper operation of the system: each player can try to generate a signature >from any subset of T broadcasted partial signatures, and then check it using the known public exponent e. If the result is a proper signature, there is no need for the verification of the partial signatures.

In the De Santis et al. [DDFY94] solution the shares $d_1, ..., d_N$ of the secret key d can be viewed themselves as RSA secret keys. Yet, for each of these keys the *public exponent*, e_i, is unknown even to

the signer, P_i. Furthermore, the public exponent *cannot* be known, as it would expose $\phi(n)$. Hence, we are faced with the problem of checking a partial signature for which we do not have a public exponent. Under the circumstances where the public exponent is not known, there must be some other information that allows the verification of such partial signatures. This information is the data $V_1, ..., V_N$, which is generated in the Dealing Phase.

Different applications which involve fault-tolerant threshold RSA warrant various properties from the RSA protocol. Some applications may require minimal communication between the players during the Signature Phase. Other applications might need that the verification will be "publicly verifiable", namely, that any person, even outside the group of N players, can verify every partial signature.

Thus we have devised two different protocols for verification of partial signatures. The choice of which protocol to use will be motivated by the needs of a specific application. The first protocol has a *non-interactive* Signature Phase, and requires each player to hold local secret verification data. The second protocol requires *interaction* in order to generate the signature, yet all the verification data is public. In the Dealing Phase of the second protocol a publicly known *sample* message and its partial signatures are generated. In the Signature Phase anyone can verify any player's partial signature by interacting with him, and using the sample signature as the basis for the proof. Both solutions are efficient computation-wise (a verification involves only a few RSA exponentiations). Surprisingly, our non-interactive verification is even more efficient in computation than the interactive one. Communication-wise the non-interactive solution is clearly optimal, while the interactive solution requires the exchange of four messages. Both protocols do not leak information that can be used by the players (even malicious ones) to forge other signatures.

A final comment on the level of trust needed from the dealer, i.e. the entity that shares the secret key and the verification data $V_1, ..., V_N$ among the players. Unfortunately it is not known how to efficiently generate in a distributed manner an RSA key in shared form [2] (indeed one could use generic results for secure multiparty computation [GMW87], but those are outside the realm of practicality). Hence, at this stage of knowledge, we need to assume that a single dealer generates the public/private RSA keys. For the non-interactive solution we always assume that the dealer is honest. In the body of the paper we describe the interactive solution, under the assumption that the dealer is trusted, yet in the appendix (see Appendix B) we present methods to verify that the dealer has acted properly in the Dealing Phase. The verification of the dealer's actions has applications to key-escrow systems.

Organization of the paper: Section 3 describes the model and the threshold-RSA solution of [DDFY94]. Our non-interactive solution is described in Section 4. The interactive protocol appears in Section 5.

3 Preliminaries

MODEL. We assume that our computation model is composed of a set of N *players* $\{P_1, \ldots, P_N\}$. They are connected by a complete network of private (i.e. untappable) point-to-point channels. In addition, they have access to a dedicated broadcast channel; by dedicated we mean that if P_i broadcasts a message, it will be recognized by the other players as coming from P_i.

These assumptions (privacy of the communication channels and dedication of the broadcast channel) allow us to focus on a high-level description of the protocols. However, it is worth noticing that these abstractions can be substituted with standard cryptographic techniques for privacy and authentication.

[2] Such results are known for El-Gamal like based signature schemes [Fel87, Ped91, GJKR96].

THE ADVERSARY. We assume that an adversary, $\mathcal{A}$, can corrupt up to $T-1$ of the N players in the network. We consider the worse possible kind of adversary, i.e. a *malicious* adversary that learns all the information held by the corrupted players and hears the broadcasted messages. He may cause corrupted players to behave in *any* (possibly malicious) way. We assume that the computational power of the adversary is adequately modeled by a probabilistic polynomial time Turing machine.

3.1 Threshold Sharing of RSA functions

In [DDFY94] the authors show how to perform threshold sharing of RSA functions. The main problem is that it is not possible to use Shamir's (T, N)-threshold secret sharing [Sha79], as the secret key d belongs to the ring $[\phi(n)]$, while we need a field in order to perform polynomial interpolation.

De Santis et al. [DDFY94] solve this problem by extending Shamir's scheme to work over any Abelian group $(G, +)$. A sketch of their work is presented here, but readers are referred to [DDFY94] for details. For our purposes let G be $Z_{\phi(n)}$ and H be Z_n^*, where n is an RSA modulus. Let $\pi > N$ be a small prime and let u be a root of the cyclotomic polynomial $p(x) = \sum_{j=0}^{\pi-1} x^j$. Consider the ring $G[u] \equiv G[x]/(p(x)) \equiv G^{\pi-1}$. Elements in this ring are $(\pi-1)$-vectors of elements in G. Addition and multiplication are defined using the isomorphism to $G[x]/(p(x))$. If one defines $x_i \stackrel{\text{def}}{=} \sum_{j=0}^{i-1} u^j$, it turns out that all x_i's and $(x_i - x_j)$'s have (multiplicative) inverses in $G[u]$.

The extended Shamir scheme is then as follows: the dealer chooses a random polynomial R over the ring $G^{\pi-1}$ of degree $T-1$ such that $R(0) = [d, 0, \ldots, 0]$, and gives $d_i = R(x_i)$ to player P_i. Because of the properties of the x_i's it is then possible to perform polynomial interpolation from at least T shares to reconstruct $[d, 0, \ldots, 0] = \sum_j \lambda_j d_j$ where λ_i are the appropriate Lagrange coefficients.

However, in our case we need to recover $m^d \bmod n$ instead of d. If $d_i = [d_{i,1}, \ldots, d_{i,(\pi-1)}] \in G^{\pi-1}$ then P_i broadcasts $S_i = [m^{d_{i,1}}, \ldots, m^{d_{i,(\pi-1)}}] \in H^{\pi-1}$. Given T of the shares S_j, one can compute $S = \prod_j S_j^{\lambda_j}$, where multiplication and exponentiation in $H^{\pi-1}$ are defined as the natural extensions of addition and multiplication in $G^{\pi-1}$, respectively. The signature $m^d \bmod n$ will be the first component of the vector S.

The correctness of the [DDFY94] result relies heavily on the correctness of each partial signature.

Observation 1: The partial signature of player P_i in this scheme is just a vector of $\pi-1$ "regular" RSA signatures. Hence, it will suffice to check each of these signatures on its own. This will allow us in the following to perform operations simply modulo n and not in the algebraic structure described above. Thus, the extension of our protocols to the [DDFY94] construction is straightforward.

Observation 2: We will assume that a partial signatures is good, if it is correct up to a sign. That is, if P_i should broadcast $S_i = m^{d_i} \bmod n$, we will allow him to broadcast either S_i or $-S_i \bmod n$. This might result in the final signature being "correct up to a sign", but this can be easily checked, by testing both signatures using the public exponent e.

3.2 Notation

For a positive integer k we denote $[k] \stackrel{\text{def}}{=} \{1, \cdots, k\}$. The public modulus is denoted n. We assume $n = pq$, and $p = 2p' + 1$, $q = 2q' + 1$, where $p < q$ and p, q, p', q' are all prime numbers. Z_n^* denotes the multiplicative group of integers modulo n, and $\phi(n) = (p-1)(q-1)$ the order of this group. For an element $w \in Z_n^*$ we denote by $ord(w)$ the order of w in Z_n^* and by $ind(w)$ the index of w in this group (it holds that $ind(w) = \phi(n)/ord(w)$). The subgroup generated by an element $w \in Z_n^*$ is denoted by $<w>$. The number $d \in [\phi(n)]$ denotes the (private) signature exponent. For any message

$m \in Z_n^*$ we denote by S_m the corresponding signature on m, namely, $S_m = m^d \bmod n$.

4 Non-Interactive Robust Threshold RSA

Our non-interactive solution, is enabled by an extension of the Information Checking Protocol (ICP) to a general integer domain. The original ICP [Rab94, RB89] works only over prime fields.

A very rough description of the verification method is the following: assume two players P and V. The prover P holds values d_P (the secret key) and y. The verifier V holds b, c, such that $y = bd_P + c$. Notice that the previous equation must hold over the integers and not modulo $\phi(n)$.[3] Given a message m, the prover generates the partial signature $m^{d_P} \bmod n$, and $m^y \bmod n$. He gives these values to V, who verifies the partial signature by checking whether $(m^{d_P})^b m^c \bmod n = m^y$.

Section 4.1 exhibits the extensions of ICP. In Section 4.2 we describe the dealing of verification information. The verification of partial signatures appears in Section 4.3.

4.1 Extensions of Information Checking

The following protocol is carried out by three players: a dealer D, who is non-faulty, and two additional players: prover P and verifier V, who can be either faulty or not. In Figure 1 we present the ICP-Generation protocol over the integers, carried out by the dealer.

ICP-Gen-Integers

Input: RSA composite n, secret value $d_P \in [\phi(n)]$ known to D,
security parameters $0 \leq \delta_1, \delta_2 \leq 1$.

1. Choose $b \in [n^{\delta_1}]$ and $c \in [n^{1+\delta_1+\delta_2}]$ with uniform distribution.
2. Compute $y = c + bd_P$ over the integers.
3. Hand d_P and y to the prover P.
4. Hand b and c to the verifier V.

Figure 1: The ICP Generation Protocol

The variables y, d_P and b, c can be later used by players P and V in the following way: When the prover P wants to prove to the verifier V that he holds the value d_P which he received from D, he sends d_P and y to V. Upon receiving values $\hat{d_P}, \hat{y}$ from P, the verifier concludes that $\hat{d_P} = d_P$ only if $\hat{y} = b\hat{d_P} + c$.
For the following we denote $\mathcal{Y} \stackrel{\text{def}}{=} [n^{\delta_1+1}, \ldots, n^{1+\delta_1+\delta_2}]$.

Lemma 1 *Given values d_P, and $y \in \mathcal{Y}$, for every possible value of b there is exactly one possible value for $c \in [n^{1+\delta_1+\delta_2}]$ such that $y = bd_P + c$.*

[3] Indeed if the equation were modulo $\phi(n)$ it would be enough for P and V to combine their information and compute a multiple of $\phi(n)$ which allows an efficient factorization of n.

Proof Since the computation is over the integers, there is exactly one value of c for each b, d_P and y. Furthermore, if $b \in [n^{\delta_1}]$ and $y \in [n^{\delta_1+1}, \ldots, n^{1+\delta_1+\delta_2}]$ then value $c = y - bd_P$ is contained in $[n^{1+\delta_1+\delta_2}]$ because

$$1 \leq n^{1+\delta_1} - n^{\delta_1}\phi(n) = y_{min} - b_{max}d_{P_max} \leq c \leq y_{max} - b_{min}d_{min} = n^{1+\delta_1+\delta_2} - 1$$

which proves the lemma. ■

Lemma 2 *Pr*($y \notin \mathcal{Y}$) $\leq \frac{1}{n^{\delta_2}}$.

Proof The number of options to choose different pairs of b and c is $n^{1+\delta_1+\delta_2}n^{\delta_1}$. The range $\mathcal{Y}$ is of size $n^{1+\delta_1+\delta_2} - n^{1+\delta_1}$. From Lemma 1 it follows that each value y in this range can be generated by n^{δ_1} pairs (b, c). Consequently, the probability of y falling outside of this range is $1 - \frac{n^{\delta_1}(n^{1+\delta_1+\delta_2} - n^{1+\delta_1})}{n^{\delta_1}n^{1+\delta_1+\delta_2}} = \frac{1}{n^{\delta_2}}$. ■

Lemma 3 – ICP over the Integers.

Completeness. *If P and V follow the protocol then V always accepts d_P.*

Zero-knowledge. *Given b, c the verifier learns no additional information on the value d_P.*

Soundness. *The probability that P generates $\hat{d_P}, \hat{y}$ such that $c + b\hat{d_P} = \hat{y}$ when in fact $\hat{d_P} \neq d_P$ is at most $\frac{1}{n^{\delta_1}} + \frac{1}{n^{\delta_2}}$. We denote this occurrence as OC_1.*

Proof

Completeness. Immediate.

Zero-knowledge. Values b and c are uniformly distributed and randomly chosen without any correlation to d_P, and hence reveal no information of its value.

Soundness. Notice that Pr(OC_1) $\leq$ Pr($OC_1 \mid y \in \mathcal{Y}$) + Pr($y \notin \mathcal{Y}$). >From Lemma 1 it follows that if $y \in \mathcal{Y}$ then P will be able to generate values $\hat{d_P}, \hat{y}$ (where $\hat{d_P} \neq d_P$) which satisfy the equation $\hat{y} = b\hat{d_P} + c$ with probability at most $\frac{1}{n^{\delta_1}}$. Lemma 2 gives us the probability that y falls out of this range. Combining these probabilities we prove our lemma.■

4.2 Generation of Verification Data (Dealing Phase)

In order to generate the data for verification of partial signatures within the context of the Dealing Phase, the dealer simply runs the ICP-Gen-Integers (Figure 1) for every pair of players P_i and P_j. All these invocations have as input the same RSA composite n and security parameters δ_1, δ_2. For the invocation where P_i is the prover P, and P_j is the verifier V, the secret key input is d_i, namely P_i's secret partial key.[4]

It will be seen later that a single pair of values b, c suffices for V to verify polynomially many different signatures. This results in an efficient protocol for the dealer, as he invokes a constant number

[4] As pointed out in Section 3.1 we can regard d_i as a single element in $[\phi(n)]$, and carry out the computations accordingly.

of ICP-Gen-Integers protocols, once and for all, regardless of the number of signatures that the system will need to generate.
As a result of the complete Dealing Phase, player P_i holds the following values:
1. His share d_i.
2. Auxiliary authentication values $y_{i,1}, \ldots, y_{i,N}$ where $y_{i,j} \in \mathbb{Z}$ is used to authenticate his partial signature to P_j.
3. Verification values $b_{1,i}, \ldots, b_{N,i}$ and $c_{1,i}, \ldots, c_{N,i}$ where $b_{j,i} \in [n^{\delta_1}]$ and $c_{j,i} \in [n^{1+\delta_1+\delta_2}]$ are used to verify the correctness of P_j's partial signature.

4.3 Partial Signatures Verification (Signature Phase)

We show the protocol for verification of a partial signature where there are two players, P and V, each holding the data which they received in ICP-Gen-Integers. The protocol appears in Figure 2. In the context of the Signature Phase this protocol will be carried out by every pair of players. After executing all these invocations of the Non-interactive Verification Protocol, player P_i will take a subset of T shares which he has accepted, and will generate the signature for m.

Non-interactive Verification

Input: Player V: $b \in [n^{\delta_1}]$, $c \in [n^{1+\delta_1+\delta_2}]$
Player P: $d_P \in [\phi(n)]$, $y = bd_P + c$
Both players: message $m \in Z_n^*$, RSA composite n

1. P broadcasts the partial signature $S = m^{d_P} \mod n$ and the auxiliary value $Y = m^y \mod n$
2. V checks if $S^b m^c = Y$. If yes, he concludes that $S = \pm m^{d_P} \mod n$ and accepts it.

Figure 2: The Non-interactive Verification Protocol

Lemma 4 Non-interactive Verification *Assume that a cheating prover P^* cannot break RSA (in particular, he does not know and cannot compute the factorization of n). Let $n = pq$, where $p < q$, $p = 2p' + 1$, $q = 2q' + 1$, and p, q, p', q' are all prime numbers.*

Completeness. *If P and V follow the protocol then V always accepts the partial signature.*

Soundness. *A cheating prover P^* can convince V to accept $S \neq \pm m^{d_P} \mod n$, with probability at most $\frac{1}{p'} + \frac{1}{n^{\delta_1}} + \frac{1}{n^{\delta_2}}$.*

Zero-knowledge. *Any (possibly cheating) verifier V^* interacting with prover P does not learn any information beyond the signature $S = m^{d_P} \mod n$.*

Proof

Completeness. Immediate.

Soundness. First we shall examine the case where $y \in \mathcal{Y}$. Notice that the verifier uses a deterministic procedure to accept or reject the published pair S, Y, hence the probability stated in the lemma, is taken over the coin-tosses of the dealer in ICP-Gen-Integers which are consistent with the view of the prover (i.e. the value y). In order for P to convince V to accept S, Y, it must hold that $Y = S^b m^c \mod n$. We know from the ICP-Gen-Integers protocol that $y = bd_P + c$, and hence $m^y = (m^{d_P})^b m^c \mod n$. By dividing these two equations we get that:

$$Y m^{-y} = (S m^{-d_P})^b \mod n \tag{1}$$

This means that $Y m^{-y}$ must be in the subgroup $< S m^{-d_P} >$. Let k be the minimal value such that $Y m^{-y} = (S m^{-d_P})^k \mod n$. Consequently, Equation (1) is satisfied only if: $b = k \mod ord(S m^{-d_P})$. Since b is chosen at random with uniform distribution from $[n^{\delta_1}]$, the probability that a pair (S, Y) satisfies Equation 1 is

$$\Pr(\, b = k \mod ord(S m^{-d_P})\,) \leq \frac{\left\lceil \frac{n^{\delta_1}}{ord(S m^{-d_P})} \right\rceil}{n^{\delta_1}} \leq \frac{1}{ord(S m^{-d_P})} + \frac{1}{n^{\delta_1}}.$$

There are only four elements of Z_n^* whose order is smaller than p', namely the four roots of unity. If $S m^{-d_P} = \pm 1 \mod n$ then $S = \pm m^{d_P} \mod n$. If the prover finds S such that $S m^{-d_P}$ is a non-trivial root of unity, then he can factor n. For all other choices of S, order $ord(S m^{-d_P}) \geq p'$. This completes the proof for the case where $y \in \mathcal{Y}$. In addition, from Lemma 2 we know that the probability that $y \notin \mathcal{Y}$ is at most $\frac{1}{n^{\delta_2}}$, by combining these two probabilities we get the desired probability.

Zero-knowledge. Values b and c are picked independently from d_P, hence they give out no information on d_P. Furthermore, knowing b, c and $S = m^{d_P} \mod n$, the verifier can compute $Y = m^y = S^b m^c \mod n$.■

5 Interactive Robust Threshold RSA

The basic idea underlying the interactive solution is that it suffices to know a single sample message and its partial signature (w, w^{d_P}) to verify the partial signature of any other message m, under the same key. The sample message will be used in the extension of a protocol by Chaum's and van Antwerpen's [CA90, Cha90, BCDP91].

In this section we assume that the dealer is trusted throughout the Dealing Phase, in Appendix B we show how to dispense with this assumption.

5.1 Generation of Verification Data (Dealing Phase)

The dealing of the verification information during the initialization protocol consists of choosing a public *sample* message w and generating sample partial signatures w^{d_i}, for each one of the partial keys held by the players.

5.2 Verification of Partial Signatures (Signature Phase)

The protocol of Chaum and van Antwerpen [CA90], which was further developed in [Cha90, BCDP91], is designed to prove in zero-knowledge the equality of the discrete logarithms of two elements relative

Sample-Signature-Generation

Input: Public: RSA modulus n
Dealer $\mathcal{D}$: key-shares $d_i \in [\phi(n)]$, for $i = 1, 2, \ldots, N$

1. $\mathcal{D}$ chooses a random value w in Z_n^* and broadcasts values $w_i = w^{d_i} \bmod n$ for $i = 1, 2, \ldots, N$.

Figure 3: The Sample Signature Generation Protocol

to two different bases, over a prime field Z_p. The protocol and the proof presented in the above papers do not work over Z_n^* for a composite n as required here, since they strongly rely on the existence of a generator for the multiplicative group Z_p^*.

For clarity of presentation, we concentrate on two players only, the prover P and the verifier V. The player P has his secret key $d_P \in [\phi(n)]$ and both players have access to a publicly known sample message w and its partial signature, $w^{d_P} \bmod n$, under P's key. For any $x \in Z_n^*$ we denote by S_x the corresponding signature on x, namely, $S_x = x^{d_P} \bmod n$. By $\hat{S}_m$ we denote the "alleged" signature on m, i.e. a string claimed (but not yet verified) to be the signature of m.

Interactive Verification

Input: Prover: secret $d_P \in [\phi(n)]$
Common: RSA composite n, sample message $w \in Z_n^*$,
signature S_w, message $m \in Z_n^*$, claimed $\hat{S}_m$

1. V chooses $i, j \in_R [n]$ and computes $R \stackrel{\text{def}}{=} m^i w^j \bmod n$
   $V \longrightarrow P : R$
2. P computes $S_R \stackrel{\text{def}}{=} R^{d_P} \bmod n$
   $P \longrightarrow V : S_R$
3. V verifies that $S_R = \hat{S}_m^i {S_w}^j \bmod n$.
   If equality holds then V concludes that $\hat{S}_m = \pm m^{d_P} \bmod n$ and accepts it.

Figure 4: The Interactive Verification Protocol

We state the security properties of the Interactive Verification protocol in Lemma 5 below. The proof of this lemma appears in Appendix A.

Lemma 5 Interactive Verification. *Assume that a cheating prover P^* cannot break RSA (in particular, he does not know and cannot compute the factorization of n), and that w was chosen at random. Let $n = pq$, where $p < q$, $p = 2p' + 1$, $q = 2q' + 1$, and p, q, p', q' are all prime numbers.*

Completeness. *If P and V follow the protocol then V always accepts.*

Soundness. *No cheating prover P^* can convince V to accept $\hat{S}_m \neq \pm m^{d_P} \bmod n$, except for a negligible probability.*

Zero-knowledge. *Any (possibly cheating) verifier V^* interacting with prover P does not learn any information beyond the signature $S_m = m^{d_P} \bmod n$.*

The protocol in figure 4 is actually not *zero-knowledge*. For example, a cheating verifier V^* can choose R in a different way than specified and then learn S_R, which V^* could not compute by himself. However, there are well-known techniques [GMW86, BCC88, Gol95] to add the zero-knowledge property to the above protocol using the notion of a *commitment function*: Instead of P sending S_R in Step 2, he sends a commitment $commit(S_R)$, after which V reveals to P the values of i and j. After checking that $R = m^i w^j \bmod n$, P sends S_R to V. The verifier checks that S_R corresponds to the value committed by P and then performs the test of Step 4 above.

The zero-knowledge condition is achieved through the properties of the commitment function, namely, (I) $commit(x)$ reveals no information on x, and (II) P cannot find x' such that $commit(x) = commit(x')$. Commitment functions can be implemented in many ways. For example, in the above protocol $commit(S_R)$ can be implemented as RSA encryption of S_R concatenated with a random string r (the encryption is required to be semantically secure, see [GM84, BR94]), using a public key for which the private key is not known to V (and possibly, not even known to P). To open the commitment, P reveals both S_R and r. This implementation of a commitment function is very efficient as it involves no long exponentiations.

6 Conclusions and Further Applications

We present two protocols for verifying partial signatures. The first protocol is a non-interactive one, the second is interactive, yet provides the ability to have public verification of the partial signatures. Both protocols are low on computation and communication. Thus, achieving an efficient, robust, threshold-RSA signature scheme.

Our techniques are closely related to the notion of undeniable signatures [CA90], and can form the basis for RSA-based undeniable signatures (known undeniable signature schemes are based on exponentiation-based systems, not on RSA). Undeniable signatures are characterized by the fact that public information is not sufficient in order to verify the signature but interaction with the signer is required for such verification. The techniques we present can be applied to separate between the signing and verification processes in the sense that a signer could delegate the ability to verify signatures to a third party while the latter cannot forge signatures.

We mention again the applicability of our results to key-escrow systems, in particular the verification of proper conduct on the part of the dealer.

References

[BCC88] G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *JCSS*, 37(2):156–189, 1988.

[BCDP91] J. Boyar, D. Chaum, I. Damgård, and T. Pedersen. Convertible undeniable signatures. In A.J. Menezes and S. A. Vanstone, editors, *Proc. CRYPTO 90*, pages 189–205. Springer-Verlag, 1991. Lecture Notes in Computer Science No. 537.

[BK89] M. Blum and S. Kannan. Program correctness checking and the design of programs that check their work. In *Proc. of the 21st ACM Symposium on Theory of Computing*, 1989.

[Boy86] C. Boyd. Digital Multisignatures. In H. Baker and F. Piper, editors, *Cryptography and Coding*, pages 241–246. Claredon Press, 1986.

[BR94] M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *EuroCrypt'94*, 1994.

[CA90] David Chaum and Hans Van Antwerpen. Undeniable signatures. In G. Brassard, editor, *Proc. CRYPTO 89*, pages 212–217. Springer-Verlag, 1990. Lecture Notes in Computer Science No. 435.

[Cha90] D. Chaum. Zero–knowledge undeniable signatures. In *Proc. EUROCRYPT 90*, pages 458–464. Springer-Verlag, 1990. Lecture Notes in Computer Science No. 473.

[DDFY94] Alfredo De Santis, Yvo Desmedt, Yair Frankel, and Moti Yung. How to share a function securely. In *Proc. 26th ACM Symp. on Theory of Computing*, pages 522–533, Santa Fe, 1994. IEEE.

[Des94] Yvo G. Desmedt. Threshold cryptography. *European Transactions on Telecommunications*, 5(4):449–457, July 1994.

[DF90] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In G. Brassard, editor, *Proc. CRYPTO 89*, pages 307–315. Springer-Verlag, 1990. Lecture Notes in Computer Science No. 435.

[DF92] Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures. In J. Feigenbaum, editor, *Proc. CRYPTO 91*, pages 457–469. Springer, 1992. Lecture Notes in Computer Science No. 576.

[Fel87] P. Feldman. A Practical Scheme for Non-Interactive Verifiable Secret Sharing. In *Proceeding 28th Annual Symposium on the Foundations of Computer Science*, pages 427–437. IEEE, 1987.

[FGY96] Y. Frankel, P. Gemmell, and M. Yung. Witness-based Cryptographic Program Checking and Robust Function Sharing. To appear in proceedings of STOC96, 1996.

[GJKR96] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold dss signatures. To appear in EuroCrypt'96, 1996.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *JCSS*, 28(2):270–299, April 1984.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing*, 17(2):281–308, April 1988.

[GMR89] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. *SIAM. J. Computing*, 18(1):186–208, February 1989.

[GMW86] O. Goldreich, S. Micali, and A. Wigderson. Proofs that Yield Nothing but the Validity of the Assertion, and a Methodology of Cryptographic Protocol Design. In *Proceeding 27th Annual Symposium on the Foundations of Computer Science*, pages 174–187. ACM, 1986.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to Play Any Mental Game. In *Proceeding 19th Annual Symposium on the Theory of Computing*, pages 218–229. ACM, 1987.

[Gol95] Oded Goldreich. *Foundation of Cryptography – Fragments of a Book*. Electronic Colloquium on Computational Complexity, February 1995. Available online from `http://www.eccc.uni-trier.de/eccc/`.

[Mic92] Silvio Micali. Fair public-key cryptosystems. In Ernest F. Brickell, editor, *Proc. CRYPTO 92*, pages 113–138. Springer-Verlag, 1992. Lecture Notes in Computer Science No. 740.

[Ped91] T. Pedersen. Distributed provers with applications to undeniable signatures. In *EuroCrypt'91*, 1991.

[Rab94] T. Rabin. Robust Sharing of Secrets When the Dealer is Honest or Faulty. *Journal of the ACM*, 41(6):1089–1109, 1994.

[RB89] T. Rabin and M. Ben-Or. Verifiable Secret Sharing and Multiparty Protocols with Honest Majority. In *Proceeding 21st Annual Symposium on the Theory of Computing*, pages 73–85. ACM, 1989.

[Sha79] A. Shamir. How to Share a Secret. *Communications of the ACM*, 22:612–613, 1979.

A Proof of Lemma 5

The proof of the completeness property of the protocol is immediate. The rest of this section is devoted to the proof of soundness.

Lemma 6 *The prover's cheating probability in the Interactive Verification Protocol is at most* $\frac{ind(w)}{ord(\hat{S}_m/m_P^d)} + 2\frac{n-\phi(n)}{n}$.

Proof Denote by b the quotient $\frac{\hat{S}_m}{m_P^d}$. If $b = 1$ the signature $\hat{S}_m$ is correct. If $b \neq 1$, the prover's probability to cheat, i.e., to convince V to accept S_m, is maximized by choosing S that passes V 's test (in step 3) with maximal probability (relative to the values i, j chosen by V). Since the prover chooses S after having seen the "challenge" R from V (and based on its knowledge of $\hat{S}_m$, m, w, d_P, and n), then the proof of soundness needs to capture that some information on i, j (at least from the information theoretic point of view) is available to the prover when selecting S.

In the actual protocol, V chooses i, j randomly from the set $[n]$; for simplicity of analysis we will assume that these values are chosen from $[\phi(n)]$, and will account for the event that either i or j fall outside of this range in the prover's probability to cheat. The probability of such event (i.e., that i or $j \notin [\phi(n)]$) is at most $2\frac{n-\phi(n)}{n}$, and it appears in the lemma's bound.

Therefore we assume $i, j \in_R [\phi(n)]$ and define the following two sets $E(R) = \{(i,j) \in [\phi(n)] : R = m^i w^j \mod n\}$ and $E(R,S) = \{(i,j) \in E(R) : S = \hat{S}_m^i {S_w}^j \mod n\}$. By chosing a value S in step 2 of the protocol, the prover P can cheat V to accept $\hat{S}_m$ only if the pair (i,j) chosen by V in step 1 actually belongs to $E(R,S)$. Since the choice of V is uniformly distributed over $E(R)$, then the probability of P to succeed is at most $\frac{|E(R,S)|}{|E(R)|}$. This is true even after P has seen R, and even if P could explicitly compute all the pairs in the set $E(R)$ and $E(R,S)$. By bounding the above expression for all possible values of R and S we get a bound on the prover's probability to cheat. Therefore, the lemma follows from the following bounds (which are proven below) on the size of the above sets for *any* choice of R and S (and any value of m, w, d_P, and $\hat{S}_m$):

(A.1) $|E(R)| \geq \phi(n)$

(A.2) $|E(R,S)| \leq \frac{ind(w)\cdot\phi(n)}{ord(b)}$ ■

We stress that the lemma holds also for a computationally unbounded cheating prover. Also, the bound in the lemma is tight for such a prover (up to the term $2\frac{n-\phi(n)}{n}$). Next, we apply the lemma to prove the soundness of the protocol in the case that n is chosen with a particular structure, and the (cheating) prover cannot break RSA.

Corollary 1 *Assume the prover P cannot break RSA (in particular, it does not know and cannot compute the factorization of n), and that w was chosen at random (or as an element of small index). Let $n = pq$, where $p < q$, $p = 2p' + 1$, $q = 2q' + 1$, and p, q, p', q' are all prime numbers. Then, the prover's probability to convince V of accepting $\hat{S}_m \neq \pm m^{d_P} \mod n$ is at most $\frac{O(1)}{p}$.*

Proof The bound in Lemma 6 is given in terms of the order of some elements in the group Z_n^*. Thus, in order to establish the exact bound for the above special form of n we need to study the order of elements in this particular group. There is one element of order 1 in Z_n^* (the unit element), 3 of order 2 (-1 and two other non-trivial roots of 1), $4p' + 4q' - 8$ elements of order ranging between p' and $2q'$, and the rest have all order which is at least $p'q'$. (This can be argued based on the special form of p and q and the order of elements modulo these primes, and then using the Chinese Remainder Theorem). In particular, the order of w, which is chosen at random, is at least $p'q'$, with probability $1 - \frac{2(p'+q')}{\phi(n)} \geq 1 - \frac{2}{p'}$, and then $ind(w)$ which equals $\phi(n)/ord(w)$ is at most 4 (notice that $\phi(n) = 4p'q'$).

As for b, a successful cheating of P happens when it convinces V to accept a value $\hat{S}_m = bm^{d_P} \bmod n$, for $b \neq \pm 1$. Notice that the prover (who knows d_P) can compute b. This excludes the possibilty that b would be one of the non-trivial square roots of 1, since knowledge of such an element would allow the prover to factor n. Therefore, b must be of order at least p'. Finally, the expression $\frac{n-\phi(n)}{n}$ is at most $1/p'$ in this case. The corollary then follows by replacing these values in the bound expression $\frac{ind(w)}{ord(\hat{S}_m/m_P^d)} + 2\frac{n-\phi(n)}{n}$. in Lemma 6. ■

Proof of inequalities (A.1) and (A.2). We proceed to prove inequalities (A.1) and (A.2) as we need to complete the proof of Lemma 6. We need to define some more sets and prove a few lemmas (we omit the $\bmod n$ notation when it is clear from the context). We define $I(R) = \{i : \exists j, (i,j) \in E(R)\}$ and $I(R,S) = \{i : \exists j, (i,j) \in E(R,S)\}$. To prove inequality (A.1) we need to bound the size of the set $E(R)$. We do this by relating the size of $E(R)$ to that of $I(R)$ in the next lemma, and then bounding $|I(R)|$.

Lemma 7 $\forall R, |E(R)| = ind(w) \cdot |I(R)|$.

Proof Fix $i \in I(R)$. The pairs $(i,j_1),(i,j_2) \in E(R)$ iff $R = m^i w^{j_1} = m^i w^{j_2}$ iff $w^{j_1-j_2} = 1$ iff $ord(w)$ divides $j_1 - j_2$. Therefore, for each $i \in I(R)$ there exist exactly $ind(w) = \phi(n)/ord(w)$ j's such that $R = m^i w^j$. ■

Lemma 8 $\forall R, |I(R)| = 0$ *or* $|I(R)| \geq ord(w)$

Proof For any given value of R, $i_1, i_2 \in I(R)$ iff there are j_1, j_2 such that $R = M^{i_1} w^{j_1} = M^{i_2} w^{j_2}$. This implies that for any such i_1, i_2, $m^{i_1-i_2} \in < w >$. On the other hand, if for values Δ and i, $m^{\Delta} \in <w>$, and $i \in I(R)$ then $i + \Delta \in I(R)$ (because then there exist j, j' such that $R = m^i h^j$ and $m^{\Delta} = w^{j'}$ from which it folllows that $R = m^{i+\Delta} w^{j-j'}$). Therefore, we get that $I(R) = \{i_0 + \Delta : m^{\Delta} \in <w>$ and $\Delta < \phi(n)\}$ where i_0 is the minimal element in $I(R)$. Thus, if $|I(R)| \neq 0$ then $|I(R)| = |D|$, where $D = \{\Delta < \phi(n) : m^{\Delta} \in <w>\}$. We proceed to bound the size of D. Using standard arguments it is easy to show that if δ is the minimal non-zero element of D then the elements of D are exactly the multiples of δ (smaller than $\phi(n)$). Thus, $|D| = \phi(n)/\delta$. We end the proof by showing $\delta \leq ind(w)$. Let $i_1 < i_2 \leq \delta$. The cosets $m^{i_1} < w >$ and $m^{i_2} < w >$ are disjoint (a common element would imply that $w^{i_2-i_1} \in < w >$ in contradiction to the minimality of δ). Thus, $m^1 < w >$, $m^2 < w >$, $\cdots m^{\delta} < w >$, are δ disjoint cosets in Z_n^* each of size $ord(w)$. We have $\delta \leq \frac{|Z_n^*|}{ord(w)} = \frac{\phi(n)}{ord(w)} = ind(w)$. In sum, $|I(R)| = |D| = \phi(n)/\delta \geq \phi(n)/ind(w) = ord(w)$ ■

Lemma 9 $\forall R, S, |E(R,S)| \leq ind(w) \cdot |I(R,S)|$

Proof Similarly to the proof of Lemma 7 one can show that for any $i \in I(R)$, if the pairs $(i,j_1),(i,j_2) \in E(R)$ then $ord(w)$ divides $j_1 - j_2$. Therefore, for each $i \in I(R)$ there are at most $ind(w) = \phi(n)/ord(w)$ different j's for which $(i,j) \in E(R)$. ■

Lemma 10 $\forall R, S, |I(R,S)| \leq \frac{\phi(n)}{ord(b)}$, *for* $b = \frac{\hat{S}_m}{m^d}$.

Proof Let (i_1,j_1), (i_2,j_2) be elements of $E(R,S)$. By the definition of $E(R,S)$, and by putting $\Delta_i \stackrel{\text{def}}{=} i_1 - i_2$ and $\Delta_j \stackrel{\text{def}}{=} j_2 - j_1$, we get $m^{\Delta_i} = w^{\Delta_j}$ and $\hat{S}_m^{\Delta_i} = S_w^{\Delta_j}$. Then,

$$\hat{S}_m^{\Delta_i} = S_w^{\Delta_j} = (w^d)^{\Delta_j} = (w^{\Delta_j})^d = (m^{\Delta_i})^d = (m^d)^{\Delta_i},$$

which implies $b^{\Delta_i} = (\hat{S}_m/m^d)^{\Delta_i} = 1$, and then $ord(b)$ divides Δ_i. In other words, we have proved that for any $i_1, i_2 \in I(R,S)$, $ord(b)$ divides $i_1 - i_2$. Since all elements in $I(R,S)$ are smaller than $\phi(n)$ there can be at most $\frac{\phi(n)}{ord(b)}$ such elements. ■

Corollary 2 *Inequalities (A.1) and (A.2) hold, namely,* $|E(R)| \geq \phi(n)$ *and* $|E(R,S)| \leq \frac{ind(w)\cdot\phi(n)}{ord(b)}$.

Proof Directly from Lemmas 7,8, 9 and 10. ■

B Verifying the sharing of the dealer

In this section we show how to verify that the dealer performs correctly the sharing of the keys and of the verification data in the interactive protocol of Section 5. This will allow us to reduce significantly the amount of trust on the server that initializes the system. One thing to notice though is that we cannot tolerate a malicious dealer $\mathcal{D}$ that cooperates with some of the players P_i. Indeed in this case we could not avoid that $\mathcal{D}$ communicates to the bad players the factorization of n. As we saw in Section 5, the security of the signature protocol is guaranteed only in the case that bad players cannot factor n. So in this case there is nothing we can do. However we believe that the verification procedure described in this section has still some value. Indeed one can picture a scenario in which the dealer is a black-box that generates RSA keys, their sharings and the verification information. This procedure will allow to check for the correctness of the program inside the black box.

VERIFICATION OF THE PARAMETERS. We need to check that the dealer chooses the modulus n of the right form, i.e. $n = pq$ with $p = 2p' + 1$ and $q = 2q' + 1$. Although this choice can be theoretically checked using the general results of [GMW87] on secure computation, the resultant solution would be hardly practical. To alleviate this problem one could have the dealer generate a large set of moduli $n_1, n_2, \cdots$ from which the players collectively choose a random element, say n_i. Next, $\mathcal{D}$ shows the factorization into primes of all the other moduli in the set. If all are of the right form then n_i is chosen as the modulus n, otherwise $\mathcal{D}$ is disqualified. Also in order to make sure that the sample message w is chosen at random, the players collectively generate it by using some protocol for collective coin toss.

VERIFICATION OF SHARES AND SAMPLE SIGNATURES. The following is a procedure by which each player can verify the correct dealing of d into the shares $d_1, \ldots, d_N$, and the correct value of the sample signatures $w_i = w^{d_i}$. Verifying the correctness of the shares means that the values d_i all lie in a unique polynomial of degree $T - 1$ whose free coefficient is d (the secret exponent of the collective signature). The values w_i are correct if they correspond to the partial signatures w^{d_i} for the verified shares d_i. Let $x_1, \ldots, x_N$ be the values used for polynomial evaluation in the share generation procedure of DDFY, and let $x_0 = 0$. Let $a_{ij}, i = 1, 2, \ldots, T,\ \ j = 1, 2, \ldots, N$, be *interpolation coefficients* such that for any polynomial p of degree $T - 1$, and for $j = 0, 1 \ldots, N$ it holds that $p(x_j) = \sum_{i=1}^{T} a_{ij} p(x_i)$. Since a set of values $d_1, \ldots, d_N$ is a correct sharing of the value d if and only if there exists a polynomial p of degree $T - 1$ such that $d = p(0)$, and $d_j = p(x_j)$, for $j = 1, 2, \ldots, N$, then we get that a correct sharing is verified by the equations:

(C.1) $\qquad d = \sum_{i=1}^{T} a_{i0} d_i \qquad d_j = \sum_{i=1}^{T} a_{ij} d_i \qquad$ for $j = T + 1, \ldots, N$.

In our case the explicit values of all the shares d_i are not available to each player, therefore the checking of correct dealing is done using the equivalent of the above equations "in the exponent", namely, each player verifies that (remember that $w_i = w^{d_i}$):

(C.2) $\qquad w = \prod_{i=1}^{T} (w_i^{a_{i0}})^e \qquad w_j = \prod_{i=1}^{T} w_i^{a_{ij}} \qquad$ for $j = T + 1, \ldots, N$.

In order to be able to claim that the verification of (C.2) implies the correctnes of (C.1) we need to solve two problems. The first is the fact that there may be a value w_i which is not an exponent of w, i.e. there is no value t for which $w_i = w^t$ (this can be the case if both the dealer $\mathcal{D}$ and the player P_i are corrupted). The second problem is that even if the values w_i are all exponents of w, the equality in (C.2) only implies that the equality in (C.1) holds modulo $ord(w)$, which may be a problem if w is an element of low order.

To verify that the values w_i are indeed exponents of w we use the following sub-protocol. For each $i \in [N]$ the dealer $\mathcal{D}$ chooses a value $r \in_R [\phi(n)]$ and broadcasts $w' = w^r$. The players

collectively choose a random bit b. If $b = 0$, $\mathcal{D}$ broadcasts the value r, otherwise it broadcasts the value $d_i + r \bmod \phi(n)$. If $w_i \notin <w>$ then the probability that $\mathcal{D}$ provides a correct response is $1/2$. By repeating this procedure k times the probability of the dealer to cheat is 2^{-k}.

As for the problem that equalities (C.1) are verified only modulo $ord(w)$, we point out that because of the assumed form of p and q (i.e., $(p-1)/2$ and $(q-1)/2$ being prime numbers), a random element w has with overwhelming probability order $\phi(n)/2$ or $\phi(n)/4$. In the former case, the order of w is a multiple of the order of all other elements in Z_n^* and then (C.2) implies (C.1). In the case $ord(w) = \phi(n)/4$, the element $-w$ is of order $\phi(n)/2$. Therefore, one solution to the above problem is to repeat the described process for both w and $-w$.