Robust Distributed Multiplication without Interaction

Masayuki Abe

NTT Laboratories
Nippon Telegraph and Telephone Corporation
1-1 Hikari-no-oka Yokosuka-shi Kanagawa-ken, 239-0847 Japan
abe@isl.ntt.co.jp

Abstract. An optimally resilient distributed multiplication protocol that enjoys the property of non-interactivity is presented. The protocol relies on a standard cryptographic assumption and works over a complete, synchronous, untappable network with a broadcast channel. As long as no disruption occurs, each player uses those channels only once to send messages; thus no interaction is needed among players. The cost is an increase in local computation and communication complexity that is determined by the factor of the threshold.
As an application of the proposed protocol we present a robust threshold version of the Cramer-Shoup cryptosystem, which is the first non-interactive solution with optimal resilience.

Keywords: Distributed Multiplication, Round Complexity, Multi-Party Computation, Cramer-Shoup Cryptosystem

1 Introduction

Background: Practical cryptographic applications sometimes put their fundamental trust on one entity such as a trusted authority or a tamper-proof device. When such a trustee is not preferred, one practical solution is to distribute the trust among several parties in order to make the system robust against the leakage of partial secrets or Byzantine faults [20,12]. Secret sharing [24] allows a party to keep a secret in a distributed manner so that some application-dependent functions are computed in collaboration without revealing the secret. So far, much work has been devoted to the investigation of such multi-party computation, e.g., [18,2,6,23,1]. In particular, some recent works realized practical multi-party versions of public-key cryptographic functions such as decryption and signature generation,e.g., [10,15,8], key generation,e.g., [21,3,14,16] and so on. Sometimes they are called "function sharing" or "threshold cryptography."

One of the basic ingredients for all robust threshold schemes is verifiable secret sharing, e.g., [11,22]. A useful property of verifiable secret sharing schemes is that they allow to compute linear combination of shared secrets without any communication between players. Multiplication on shared secrets, however, remains a cumbersome process because it requires several interactions among players in

Michael Wiener (Ed.): CRYPTO'99, LNCS 1666, pp. 130–147, 1999.

order to keep the threshold unchanged. Since round complexity can be the dominant factor determining efficiency, especially when the protocol is implemented over a network where communication overhead can not be ignored, it is of practical interest to construct a distributed multiplication protocol that enjoys less round complexity and yet remains practical in computation and communication.

Related Work: A generic approach for evaluating any boolean circuit in a cryptographic multi-party setting was shown in [18], which assumes the use of a probabilistic encryption scheme. Although the result is theoretically significant, this approach will turn out to be inefficient if it is directly applied to construct a multi-party protocol for a particular function such as multiplication in a large field.

In [2], Ben-Or, Goldwasser and Wigdersen introduced the degree reduction technique that converts shares on a $2t$-degree polynomial into ones on a t-degree truncated polynomial. For n players, the technique tolerates less than $n/3$ corrupted players. The scheme shown in [23] tolerates less than $n/2$ faulty players and also follows this line. The scheme in [6] relies on the massive use of zero knowledge proofs. All the schemes that do not rely on computational assumptions require far too many interactions among players, even though some of them are constant-round. Beaver improved the scheme in [23] but the round complexity remains unchanged [1].

M. Rabin invented a simple (non-robust) distributed multiplication scheme in [17], which can be replaced with the previous degree reduction technique. In the information theoretic setting, Cramer et al. added robustness to Rabin's basic scheme by developing new Information Checking technique [7]. In the cryptographic setting, Gennaro et al., provided robustness by having each player invoke a four-move zero-knowledge interactive proof [17]. All players can be logically combined into a single verifier (by using an information theoretically secure bit commitment scheme and a verifiable secret sharing), and if this combination performs all zero-knowledge proofs in parallel, one can obtain a four-move protocol for distributed multiplication. The immediately thought is that the public-coin interactive zero-knowledge protocol can be turned into an non-interactive one by using Fiat-Shamir heuristics [13] at the cost of losing provable security in a standard cryptographic model. Cerecedo et al., introduced a dedicated protocol for multiplying a distributedly-generated random number and a shared secret in a construction of shared signature schemes [5]. Their scheme works non-interactively for thresholds under one third of the players, and two additional accesses to the communication channels are needed to obtain optimal resiliency. Their schemes are not information-theoretically secure, and hence leak some information about shared secrets. Indeed, the attack introduced by Gennaro et al., in [16] is applicable.

Our Contribution: We provide a distributed multiplication protocol that works in a cryptographic setting and offers the following properties:

- Non-interactivity: As long as no disruption occurs, players access private network and broadcast channel only once to send data without needing to synchronize to other players. The players agree on correctness implicitly by silence (or explicitly by broadcasting 1 bit message if needed). No zero-knowledge proof is used in the protocol.
- Provably secure in a standard model: The security can be proved under the intractability assumption of the discrete logarithm problem. We do not assume the use of shared random string or random oracles.
- Information theoretic secrecy: The shared secrets are secure against infinitely powerful adversaries.
- Optimal resiliency: Tolerate the corruption of less than half of players.

The cost of achieving these properties is an increase in computation and communication complexity. Our scheme suffers $\mathcal{O}(t)$ increase in computation and communication complexity, where t is the threshold, compared to the above mentioned four-move scheme by Gennaro et al. Yet it is comparable to the scheme by Cerecedo et al. in both computation and communication costs. More concretely, our scheme consumes about three to four times more computation and communication than that of the random number generation scheme by Pedersen [22], which is often used in threshold cryptography.

Our result will reduce the round complexity of several cryptographic applications, e.g., all variants of El Gamal signature scheme that have embedded multiplication, and threshold key generation for signature or encryption schemes that use a composite modulus like RSA.

We use our protocol to construct a robust threshold version of the Cramer-Shoup cryptosystem that withstands a minority of corrupt decryption servers. The protocol is non-interactive except for precomputation for randomizing decryption keys. A threshold Cramer-Shoup cryptosystem with optimal resiliency was described by Canetti et al. in [4] that uses four-move interactive zero-knowledge proofs. One of their variants has the non-interactivity property, but it only tolerates $\mathcal{O}(\sqrt{n})$ corrupted servers. Accordingly, we show the first optimally resilient and non-interactive robust threshold cryptosystem that is secure against adaptive chosen ciphertext attacks under standard cryptographic assumptions.

Organization: Section 2 sketches the model and our goal. Section 3 presents the proposed multiplication protocol. Proof of security is given in Section 4. Section 5 extends the protocol to remove an assumption about the inputs. In Section 6, we construct a threshold version of the Cramer-Shoup cryptosystem. Some concluding remarks are given in Section 7.

2 Model

2.1 Setting

Communication channels: Our protocol assumes a *synchronous private network* where a message is assured of being delivered in a fixed period. The network

is assumed to be secure and complete, that is, every pair of players is connected by an untappable and mutually authenticated channel. Furthermore, we assume the use of a *broadcast channel* where all players receive the same information sent from authenticated players.

Players and an adversary: Let $\mathcal{P}$ be a set of players such that $\mathcal{P} = \{1, \ldots, n\}$. Player $i \in \mathcal{P}$ is assumed to be a probabilistic polynomial-time Turing machine. Among those players, there are up to t *corrupt players* completely controlled by a *static adversary*.

Computational assumption: We use large primes p and q that satisfy $q|p-1$. G_q denotes a multiplicative subgroup of degree q in Z_p. It is assumed that solving the discrete logarithm problem in G_q is intractable. All arithmetic operations are done in Z_p unless otherwise stated.

2.2 Notations for Verifiable Secret Sharing

Our scheme uses an information theoretically secure verifiable secret sharing scheme by Pedersen [22]. Let g and h be elements of G_q where $\log_g h$ is unknown. To share secret S in Z_q, a dealer first chooses two t-degree random polynomials $f(X)$ and $d(X)$ from $Z_q[X]$ such that $f(0) = S$ is satisfied. Let R denote random free term of $d(X)$. The dealer sends a pair $(S_i, R_i) := (f(i), d(i)) \in Z_q^2$ to player i via a private channel. The dealer then broadcasts $E_k := g^{a_k} h^{b_k}$ for $k = 0, \ldots, t$ where a_k and b_k are coefficients of k-degree term of $f(X)$ and $d(X)$ respectively. Note that S and R are committed to E_0 as $E_0 = g^S h^R$. Correctness of a share (S_i, R_i) can be verified by checking the relation

$$g^{S_i} h^{R_i} = \prod_{k=0}^{t} {E_k}^{i^k}. \tag{1}$$

We may refer the right side of the above verification predicate as *verification commitment* for player i. Note that any player can compute the verification commitments without knowing the corresponding shares. Hereafter, we denote the execution of this verifiable secret sharing protocol by

$$PedVSS(S, R)[g, h] \xrightarrow{f,d} (S_i, R_i)(E_0, \ldots, E_t).$$

Polynomials put on an arrow may be omitted if no misunderstanding is expected.

Secret S is reconstructed by interpolation as in

$$S := \sum_{i \in \mathcal{Q}} \lambda_{i,\mathcal{Q}} S_i \bmod q, \quad \text{where} \quad \lambda_{i,\mathcal{Q}} := \prod_{j \in \mathcal{Q}, j \neq i} \frac{j}{j-i} \bmod q \tag{2}$$

where $\mathcal{Q}$ is any set of at least $t+1$ qualified players whose share (S_i, R_i) satisfies Equation 1.

Lemma 1. *The above verifiable secret sharing scheme offers the following properties:*

1. *If all players follow the protocol, shares are accepted with probability* 1.
2. *For any* $\mathcal{Q} \subseteq \mathcal{P}$ *of size no less than* $t+1$*, a set of shares* $(S_i, R_i) \mid i \in \mathcal{Q}$ *each of which satisfies Equation 1 recovers* (S, R) *that satisfies* $E_0 = g^S h^R$ *with overwhelming probability in* $|q|$.
3. *Let* $view_{\mathcal{A}}$ *be a unified view in a protocol run obtained by corrupted players in* $\mathcal{A}$*. For any* $\mathcal{A} \subset \mathcal{P}$ *of size at most* t*, for any secret* $S \in Z_q$*, and for any* $x \in Z_q$*,*

$$Prob[S = x | view_{\mathcal{A}}] = Prob[S = x].$$

As it is well-known, the players can share a random number unknown to any set of players less than $t+1$ by using *PedVSS* . We refer to this shared random number generation protocol as

$$RandVSS([S], [R])[g, h] \rightarrow (S_i, R_i)(E_0, \ldots, E_t).$$

Brackets in parentheses imply that the numbers are unknown to any set of players less than the threshold.

2.3 Goal

Our goal is to construct a protocol such that players in $\mathcal{P}$ eventually share the product of $A \in Z_q$ and $B \in Z_q$ as if an honest dealer executes

$$PedVSS(A \cdot B,\ R^c)[g, h] \rightarrow (C_i, R_i^c)(EC_0, \ldots, EC_t) \tag{3}$$

for some random number $R^c \in Z_q$. We assume that A and B have been already shared by an honest dealer as

$$PedVSS(A, R^a)[g, h] \rightarrow (A_i, R_i^a)(EA_0, \ldots, EA_t) \tag{4}$$
$$PedVSS(B, R^b)[g, h] \rightarrow (B_i, R_i^b)(EB_0, \ldots, EB_t) \tag{5}$$

where R^a and R^b are random numbers taken uniformly from Z_q.

3 Protocol

The main idea to reduce round complexity is to use VSS instead of ZKP to confirm the relationship of committed values. Recall that if a dealer completes $PedVSS(S, R) \rightarrow (S_i, R_i)(E_0, \ldots, E_t)$ correctly, it implies that the dealer knows S and R committed to $E_0 = g^S h^R$. Conversely, if the dealer does not know the representation of E_0, he can not complete the protocol successfully. Similarly, if P_i completes two *PedVSS* executions such as

$$PedVSS(S, R) \rightarrow (S_i, R_i)(E_0, \ldots, E_t),$$
$$PedVSS(S', R') \rightarrow (S_i', R_i')(E_0', \ldots, E_t')$$

and if $S_i = S'_i$ holds for at least $t+1$ evaluation points, it proves equality of committed values, that is, $S = S'$. Those arguments are derived immediately from Lemma 1. Furthermore, if P_i completes three executions,

$$PedVSS(S, R) \rightarrow (S_i, R_i)(E_0, \ldots, E_t),$$
$$PedVSS(S', R') \rightarrow (S'_i, R'_i)(E'_0, \ldots, E'_t),$$
$$PedVSS(S'', R'') \rightarrow (S''_i, R''_i)(E''_0, \ldots, E''_t),$$

and $S_i + S'_i = S''_i$ holds for more than $t+1$ i-s, it implies additive relation of committed values, that is, $S+S' = S''$. Multiplicative relation, which is our main interest, can be proved in a rather tricky way as is shown in the main protocol.

Our protocol is constructed over the simplified multiplication protocol in [17], which is reviewed below. This protocol works with t up to $t < n/2$.

[**Protocol:** Basic Multiplication]

BM-1: Each player i picks a t-degree random polynomial to share $A_i \cdot B_i$. The share C_{ij} is privately sent to player j, $j \in \mathcal{P}$.

BM-2: Each player j computes his share C_j for $A \cdot B$ as

$$C_j := \sum_{i \in \mathcal{P}} \lambda_{i,\mathcal{P}} C_{ij} \bmod q.$$

[**End**]

Let $\mathcal{Q}_2$ and $\mathcal{Q}_1$ be subsets of $\mathcal{P}$ whose sizes are $2t+1$ and $t+1$ respectively. Recall that $A \cdot B$ can be recovered from $A_i \cdot B_i$ as $A \cdot B = \sum_{i \in \mathcal{Q}_2} \lambda_{i,\mathcal{Q}_2} A_i \cdot B_i$. Since the above protocol allows players to recover $A_i \cdot B_i$ by computing $A_i \cdot B_i = \sum_{j \in \mathcal{Q}_1} \lambda_{j,\mathcal{Q}_1} C_{ij}$, it holds that $A \cdot B = \sum_{i \in \mathcal{Q}_2} \lambda_{i,\mathcal{Q}_2} \sum_{j \in \mathcal{Q}_1} \lambda_{j,\mathcal{Q}_1} C_{ij} = \sum_{j \in \mathcal{Q}_2} \lambda_{j,\mathcal{Q}_2} C_j$. Therefore, $A \cdot B$ can be recovered from any set of more than $t+1$ C_j-s.

To add robustness to the basic multiplication protocol, each player must convince other players that C_{ij} is a share of $A_i \cdot B_i$. Let the verification commitment for (A_i, R_i^a) and (B_i, R_i^b) be $VA_i := \prod_{k=0}^{t} EA_k{}^{i^k}$ and $VB_i := \prod_{k=0}^{t} EB_k{}^{i^k}$ respectively.

[**Protocol:** Robust Distributed Non-interactive Multiplication]

DM-1: Each player i randomly picks t-degree polynomials f_1, d_1 and d_2 from $Z_q[X]$ so that $f_1(0) = A_i$ and $d_1(0) = R_i^a$ are satisfied. Let R_i^{ab} denote a randomly chosen free term of $d_2(X)$. Player i shares A_i twice as

$$PedVSS(A_i, R_i^a)[g, h] \quad \stackrel{f_1, d_1}{\rightarrow} (A_{ij}, R_{ij}^a)(\langle VA_i \rangle, EA_{i1}, \ldots, EA_{it}), \text{ and}$$
$$PedVSS(A_i, R_i^{ab})[VB_i, h] \stackrel{f_1, d_2}{\rightarrow} (\langle A_{ij} \rangle, R_{ij}^{ab})(EAB_{i0}, \ldots, \text{it EAB}_{it}).$$

Angle brackets mean that the variable can be locally computed by the receivers, or it has been sent before. Note that $VB_i(= g^{B_i} h^{R_i^b})$ is used as the base of the commitments in the second sharing. Player i then selects two random polynomials $f_2(X)$ and $d_3(X)$ that satisfy $f_2(0) = A_i \cdot B_i \bmod q$ and $d_3(0) = R_i^b \cdot A_i + R_i^{ab} \bmod q$, and performs

$$PedVSS(A_i \cdot B_i, R_i^b \cdot A_i + R_i^{ab})[g,h] \xrightarrow{f_2,d_3} (C_{ij}, R_{ij}^c)(\langle EAB_{i0}\rangle, EC_{i1}, ..., EC_{it}).$$

DM-2: Each player j verifies everything received from player i as

$$g^{A_{ij}} h^{R_{ij}^a} = VA_i \prod_{k=1}^{t} {EA_{ik}}^{j^k}, \tag{6}$$

$${VB_i}^{A_{ij}} h^{R_{ij}^{ab}} = \prod_{k=0}^{t} {EAB_{ik}}^{j^k}, \tag{7}$$

$$g^{C_{ij}} h^{R_{ij}^c} = EAB_{i0} \prod_{k=1}^{t} {EC_{ik}}^{j^k}. \tag{8}$$

If a check fails, player j declares so and goes to the disqualification protocol (see below).

DM-3: Let $\mathcal{I}$ be a set of qualified players in **DM-2**. $|\mathcal{I}| \geq 2t+1$ must hold. Each player j in $\mathcal{I}$ computes

$$C_j := \sum_{i \in \mathcal{I}} \lambda_{i,\mathcal{I}} C_{ij} \bmod q, \tag{9}$$

$$R_j^c := \sum_{i \in \mathcal{I}} \lambda_{i,\mathcal{I}} R_{ij}^c \bmod q, \tag{10}$$

$$EC_k := \prod_{i \in \mathcal{I}} {EC_{ik}}^{\lambda_{i,\mathcal{I}}} \text{ for } k = 0, \ldots, t, \tag{11}$$

where $EC_{i0} = EAB_{i0}$.

[**End**]

Suppose that player j finds that the shares sent from player i do not satisfy all of the verification equations in **DM-2**. All players then perform the following protocol.

[**Protocol:** DISQUALIFICATION]

DQ-1: Player i is requested to broadcast all the data that he privately sent to player j, which is $(A_{ij}, R_{ij}^a, R_{ij}^{ab}, C_{ij}, R_{ij}^c)$. (If player i keeps silent, he is disqualified immediately.)

DQ-2: If $t+1$ or more players conclude that the shares satisfy the verification equations in **DM-2**, player j must accept the shares just published. On the other hand, if $t+1$ or more players decide that those shares are faulty, player i is disqualified.

[End]

Let $\mathcal{D}$ be a set of disqualified players. If $|\mathcal{P} \setminus \mathcal{D}| < 2t+1$ happens because of the disqualification, the remaining players in $\mathcal{P} \setminus \mathcal{D}$ have to recover (A_i, R_i^a) and (B_i, R_i^b) owned by a disqualified player i. For this to be done, we use the *share recovery* protocol which has been used in proactive secret sharing [19]. Note that the protocol can be completed in a robust and secure way if at least $t+1$ honest players exist. (A_i and R_i^a can be recovered directly from shares (A_{ij}, R_{ij}^a) if more than $t+1$ correct shares exist.)

4 Security

Lemma 2 (Correctness). *If all players follow the protocol, every player i in $\mathcal{P}$ obtains a share (C_i, R_i^c) and commitments $(EC_0, \ldots, EC_t)$ that satisfy $g^{C_i} h^{R_i^c} = \prod_{k=0}^{t} {EC_k}^{i^k}$ and $A \cdot B = \sum_{i \in \mathcal{Q}} \lambda_{i,\mathcal{Q}}\, C_i \bmod q$ for any $\mathcal{Q} \subseteq \mathcal{P}$ of size no less than $t+1$.*

Proof. According to Lemma 1, any set of no less than $t+1$ correct (A_{ij}, R_{ij}^a) that passes verification 6 recovers (A_i, R_i^a) that satisfies $g^{A_i} h^{R_i^a} = VA_i$. Similarly, any set of no less than $t+1$ correct (A_{ij}, R_{ij}^{ab}) that passes verification 7 recovers $(\tilde{A}_i, R_i^{ab})$ that satisfies $VB^{\tilde{A}_i} h^{R_i^{ab}} = EAB_{i0}$. Because both A_i and $\tilde{A}_i$ are recovered from the same shares A_{ij}, we have $\tilde{A}_i = A_i$. Therefore,

$$EAB_{i0} = VB_i^{A_i} h^{R_i^{ab}} = g^{A_i \cdot B_i} h^{A_i \cdot R_i^b + R_i^{ab}}.$$

Observe that EAB_{i0} is also used to verify (C_{ij}, R_{ij}^c) with g and h. Hence, for any $\mathcal{Q} \subseteq \mathcal{P}$ of size no less than $t+1$, it holds that $A_i \cdot B_i = \sum_{j \in \mathcal{Q}} C_{ij} \lambda_{j,\mathcal{Q}} \bmod q$. Recall that $A \cdot B$ can be recovered from a set of correct $A_i \cdot B_i$ as $A \cdot B = \sum_{i \in \mathcal{I}} (A_i \cdot B_i)\, \lambda_{i,\mathcal{I}} \bmod q$ holds for any $\mathcal{I} \subseteq \mathcal{P}$ of size no less than $2t+1$. Thus,

$$\begin{aligned}
\sum_{j \in \mathcal{Q}} C_j \lambda_{j,\mathcal{Q}} &= \sum_{j \in \mathcal{Q}} \Big(\sum_{i \in \mathcal{I}} C_{ij} \lambda_{i,\mathcal{I}}\Big) \lambda_{j,\mathcal{Q}} \\
&= \sum_{i \in \mathcal{I}} \Big(\sum_{j \in \mathcal{Q}} C_{ij} \lambda_{j,\mathcal{Q}}\Big) \lambda_{i,\mathcal{I}} \\
&= \sum_{i \in \mathcal{I}} (A_i \cdot B_i) \lambda_{i,\mathcal{I}} \\
&= A \cdot B \pmod q
\end{aligned}$$

holds for any $\mathcal{Q} \subseteq \mathcal{P}$ of size no less than $t+1$. □

Lemma 3 (Secrecy). *Let $\mathcal{A}$ be $\mathcal{A} \subset \mathcal{P}$ such that $|\mathcal{A}| \leq t$. Let $view_{\mathcal{A}}$ be unified view of players in $\mathcal{A}$ during a protocol run with secrets (A, R^a) and (B, R^b). Similarly, let $view'_{\mathcal{A}}$ be a view during a protocol run with $(\tilde{A}, \tilde{R}^a)$ and $(\tilde{B}, \tilde{R}^b)$ that satisfy $g^{\tilde{A}} h^{\tilde{R}^a} = g^A h^{R^a}$ and $g^{\tilde{B}} h^{\tilde{R}^b} = g^B h^{R^b}$. Then, $view_{\mathcal{A}}$ and $view'_{\mathcal{A}}$ are perfectly indistinguishable.*

Proof. We prove the above statement by constructing a protocol simulator that works without knowing the secrets committed to EA_0 and EB_0. Let $\mathcal{H} := \mathcal{P} \setminus \mathcal{A}$. For simplicity, we exclude the disqualification steps for a while. The view $view_{\mathcal{A}}$ in a protocol run with secrets A, B can be divided into the following two parts:

$$\begin{aligned} view_{\mathcal{A},private} &= \{\Omega_j, A_j, R^a_j, B_j, R^b_j \mid j \in \mathcal{A}\}, \\ view_{\mathcal{A},received} &= \{EA_k, EB_k, A_{ij}, R^a_{ij}, R^{ab}_{ij}, C_{ij}, R^c_{ij}, \\ &\qquad EA_{ik}, EAB_{ik}, EC_{ik} \mid i \in \mathcal{H}, j \in \mathcal{A}, k = 0, \ldots, t\}, \end{aligned}$$

where Ω_j is a private random tape of player j.

According to Lemma 1, for any $\mathcal{A}$ of size no more than t, $view_{\mathcal{A},private}$ is independent of (A, B). Therefore, in order to prove the statement, it is sufficient to show that $view_{\mathcal{A},received}$ can be perfectly simulated. Given p, q, g, h and $\{EA_k, EB_k \mid k = 0, \ldots, t\}$, execute the following steps for all $i \in \mathcal{H}$:

SIM-1: For $j \in \mathcal{A}$, choose A_{ij} and R^a_{ij} randomly from Z_q. Then compute EA_{ik} for $k = 1, \ldots, t$ by solving the system of equations $g^{A_{ij}} h^{R^a_{ij}} = VA_i \prod_{k=1}^{t} EA_{ik}^{j^k}$ for $j \in \mathcal{A}$. (Note that this step does not compute discrete logarithms.)

SIM-2: For $j \in \mathcal{A}$, choose R^{ab}_{ij} randomly from Z_q. Also choose EAB_{i0} from G_q. Then, compute EAB_{ik} for $k = 1, .., t$ by solving ${VB_i}^{A_{ij}} h^{R^{ab}_{ij}} = \prod_{k=0}^{t} {EAB_{ik}}^{j^k}$ for $j \in \mathcal{A}$.

SIM-3: For $j \in \mathcal{A}$, choose C_{ij}, R^c_{ij} randomly from Z_q. Then, compute EC_{ik} for $k = 1, \ldots, t$ by solving $g^{C_{ij}} h^{R^c_{ij}} = EAB_{i0} \prod_{k=1}^{t} {EC_{ik}}^{j^k}$ for $j \in \mathcal{A}$.

As a result, $A_{ij}, R^a_{ij}, R^{ab}_{ij}, C_{ij}, R^c_{ij}, EA_{ik}, EAB_{ik}, EC_{ik} \mid i \in \mathcal{H}, j \in \mathcal{A}, k = 0, \ldots, t$ is obtained.

Let us examine the distribution of each variable. As A_{ij}, R^a_{ij} are randomly chosen in **SIM-1**, commitments EA_{ik} for $k = 1, \ldots, t$ distribute uniformly over the space defined by relation $g^{A_{ij}} h^{R^a_{ij}} = VA_i \prod_{k=1}^{t} {EA_{ik}}^{j^k}$ for $j \in \mathcal{A}$, which is exactly the same relation that those shares and commitments satisfy in a real protocol run. Observe that A_{ij}, R^a_{ij} for $j \in \mathcal{A}$ randomly distribute over ${Z_q}^{2t}$ in a real protocol run. Thus, A_{ij}, R^a_{ij} for $j \in \mathcal{A}$ and EA_{ik} for $k = 1, \ldots, t$ have the same distribution as those in a real protocol run. A similar observation is possible for the remaining variables in $view_{\mathcal{A},received}$.

Next consider a view in the disqualification protocol. Two cases arise: player j in $\mathcal{A}$ challenges player i in $\mathcal{H}$, and the reverse case. In the former case, challenged player i in $\mathcal{H}$ just broadcasts shares and commitments that have been already in the view of the adversary. In the latter case, every player i in $\mathcal{H}$ just returns an answer based on the results of verification predicates in **DM-2**, which are

computed only with shares and commitments sent from the challenged player. Therefore, executing the disqualification protocol contributes nothing to $view_{\mathcal{A}}$. Secrecy in the share recovery protocol holds according to Theorem 5 of [19]. □

Lemma 4 (Robustness). *Let $\mathcal{A}$ be a set of corrupted players. If $|\mathcal{A}| \leq t$ and $|\mathcal{P}| \geq 2t+1$, every player i in $\mathcal{P} \setminus \mathcal{A}$ obtains correct share (C_i, R_i^c) and commitments EC_k for $k = 0, \ldots, t$ as a result of the above protocol.*

Proof. Let $\mathcal{H} := \mathcal{P} \setminus \mathcal{A}$. Clearly, player i in $\mathcal{H}$ will not be disqualified in the disqualification protocol because they are in the majority. According to Lemma 1, any inconsistency among shares and corresponding commitments is detected with overwhelming probability. Since the sender of the inconsistent shares is requested to broadcast the correct ones, what he can do is to broadcast the correct ones, the incorrect ones, or halt. Broadcasting incorrect shares or halting results in being disqualified immediately. Observe that share recovery protocol can be completed if there are at least $t+1$ honest players. So even if all corrupted players are disqualified at **DM-2**, the remaining players can complete share recovery protocol and obtain correct shares and commitments that should have sent from the disqualified player(s). Thus, there are at least $t+1$ players who have at least $2t+1$ correct shares and commitments, which is sufficient to complete **DM-3** to obtain correct (C_j, R_j^c) and EC_k.

The above argument is correct if it is infeasible for player i to compute $\log_h VB_i$. Observe that

$$\begin{aligned} \log_h VB_i &= \log_h g^{B_i} h^{R_i^b} \\ &= B_i \log_h g + R_i^b. \end{aligned}$$

Since we assume $\log_h g$ is not known to any player, player i could compute $\log_h VB_i$ only if $B_i = 0$. However, as B is assumed to be honestly shared, $B_i = 0$ happens with probability $1/q$. Thus, the chance that player i distributes inconsistent shares remain negligible. □

5 Extension

In the previous section we made the assumption that B is shared by an honest dealer so that $B_i = 0$ happens rarely. However, B is typically generated randomly by players in threshold cryptography. If *RandVSS* is used for this purpose, adversarial player i can control B_i so as to make it zero by distributing his choices after receiving shares from all other players.

By adding one more *PedVSS* to the multiplication protocol, we can deal with the case of $B_i = 0$. Let h_0 be a randomly chosen member of G_q. The idea is to let each player i share A_i with bases $[VB_i, h_0] = [g^{B_i} h^{R_i^b}, h_0]$ instead of $[VB_i, h]$. Accordingly, even if $B_i = 0$, player i can not cheat because he does not know $\log_{h_0} VB_i$ $(= R_i^b \log_{h_0} h)$ unless $R_i^b = 0$. If both B_i and R_i^b equal 0, then other

players can detect it by finding $VB_i = 1$. The following is a brief description of the modified steps which correspond to **DM-1** and **DM-2**.

DM'-1: Each player i executes the following VSS-s.

$PedVSS(A_i, R_i^a)[g,h] \stackrel{f_1,d_1}{\rightarrow} (A_{ij}, R_{ij}^a)(\langle VA_i \rangle, EA_{i1}, \ldots, EA_{it})$

$PedVSS(A_i, R_i^{ab})[VB_i, h_0] \stackrel{f_1,d_2}{\longrightarrow} (\langle A_{ij} \rangle, R_{ij}^{ab})(EAB_{i0}, \ldots, EAB_{it})$

$PedVSS(A_i \cdot B_i, R_i^b \cdot A_i, R_i^{ab})[g,h,h_0] \stackrel{f_2,d_3,d_4}{\longrightarrow} (C_{ij}, T1_{ij}, T2_{ij})$
$(\langle EAB_{i0} \rangle, ET_{i1}, \ldots, ET_{it})$,

$PedVSS(A_i \cdot B_i, T3_i)[g,h] \stackrel{f_2,d_5}{\longrightarrow} (\langle C_{ij} \rangle, T3_{ij})(EC_{i0}, \ldots, EC_{it})$

R_i^{ab} and $T3_i$ are chosen randomly from Z_q. Intuitively, the first and second VSS commit $A_i \cdot B_i$ to EAB_{i0}. The third VSS distributes $A_i \cdot B_i$ committed to EAB_{i0}. The last VSS transforms the bases of commitment of $A_i \cdot B_i$ from $[g,h,h_0]$ to $[g,h]$. (The last VSS can be omitted depending on application. See the protocols in the next section for instance.)

DM'-2: Each player j verifies

$$
\begin{aligned}
VB_i &\neq 1, \\
g^{A_{ij}}\, h^{R_{ij}^a} &= VA_i \prod_{k=1}^{t} {EA_{ik}}^{j^k}, \\
{VB_i}^{A_{ij}}\, h_0^{R_{ij}^{ab}} &= \prod_{k=0}^{t} {EAB_{ik}}^{j^k}, \\
g^{C_{ij}}\, h^{T1_{ij}}\, h_0^{T2_{ij}} &= EAB_{i0} \prod_{k=1}^{t} {ET_{ik}}^{j^k}, \\
g^{C_{ij}}\, h^{T3_{ij}} &= \prod_{k=0}^{t} {EC_{ik}}^{j^k},
\end{aligned}
$$

for all $i \in \mathcal{P}$.

Shares for player j can be computed in the same way as shown in **DM-3** by replacing R_{ij}^c with $T3_{ij}$. This extended multiplication protocol retains the all properties stated in the previous section except for the case where honest player i is disqualified because of $VB_i = 1$, though it happens with negligible probability as long as adversary is polynomially bounded. In the case of executing *RandVSS* for generating B in the presence of infinitely powerful adversaries, honest players must not be disqualified to maintain information theoretic secrecy. So if $VB_i = 1$ happens for player i, it declares so and invokes another *PedVSS* that is combined to the result of *RandVSS* so that $VB_j \neq 1$ holds for all $j \in \mathcal{P}$.

6 Application

Robust threshold encryption is a useful tool for constructing applications like secret ballot electronic voting schemes where a single trusted party is not preferred. The first robust threshold cryptosystem provably secure against adaptive chosen ciphertext attack was presented by Shoup and Gennaro in [25], where the security was proved in the random oracle model. The Cramer-Shoup cryptosystem [9] is a provably secure (non-threshold) scheme whose security can be proved only by using a standard cryptographic assumption, i.e., intractability of the decision Diffie-Hellman problem.

In [4], Canetti and Goldwasser introduced a robust threshold version of the Cramer-Shoup cryptosystem which requires four-move ZKP to tolerate a minority of corrupt players. Here we present another construction that enjoys lower round complexity and optimal resiliency. In our scenario, unlike previous one by Canetti et al., players (decryption servers) are assumed to use the decryption result themselves. Accordingly, it is the player who has to be convinced of the correctness of the result. Such a scenario will also be applicable to quorum controlled systems.

6.1 Cramer-Shoup Cryptosystem

Let g_1 and g_2 be independently chosen elements in G_q. A decryption key is 6-tuple [1] $(x_1, x_2, y_1, y_2, z_1, z_2) \in_R Z_q^6$, and the corresponding encryption key is triple (X, Y, Z) where $X = g_1^{x_1} g_2^{x_2}, Y = g_1^{y_1} g_2^{y_2}$ and $Z = g_1^{z_1} g_2^{z_2}$. An encryption of a message, $M \in G_q$, is (u_1, u_2, v, m) that satisfies

$$u_1 = g_1^r,\ u_2 = g_2^r,\ m = M \cdot Z^r,\ c := Hash(u_1, u_2, m),\ v = X^r Y^{cr}$$

for $r \in_R Z_q$, where $Hash : \{0,1\}^* \to Z_q$ is a corrision-intractable hash function. To decrypt, compute

$$V := u_1^{x_1 + c\, y_1} u_2^{x_2 + c\, y_2}$$

and verify if $V = v$. If it is successful, the message is recovered by computing $M = m / u_1^{z_1} u_2^{z_2}$. Otherwise the encrypted message is rejected.

The security proof against adaptive chosen ciphertext attack in [9] utilizes the property that the decryption process leaks no information about each of the secret keys except for their linear relation such as $\log_{g_1} X = x_1 + x_2 \log_{g_1} g_2$.

A problem with the threshold scheme is that V must not be revealed to corrupt players if $V \neq v$. This means that players have to be convinced of $V \neq v$ without seeing V itself.

[1] The original scheme presented in [9] only uses z_1 for efficiency. Here we use z_1 and z_2 for the sake of key generation, however, it never influences the security. Indeed, [9] proves the security using a model that uses z_1 and z_2.

6.2 A Threshold Version

The underlying idea is to replace V with V' such as

$$V' := (V\, v^{-1})^w$$

where w is a one-time random number chosen from Z_q^*. Observe that $V' = 1$ holds if and only if $V = 1$ holds. Furthermore, for every $V' \in G_q$ and for every $V \in G_q$, there exists $w \in Z_q^*$ that satisfies $V' = (V\, v^{-1})^w$. Therefore, revealing V' does not leak any information about V in an information theoretic sense except when $V' = 1$.

For efficiency, decryption is combined with the verification process by replacing the above V' with $\tilde{V}$ such that

$$\begin{aligned}\tilde{V} &:= V'/u_1^{z_1}\, u_2^{z_2} \\ &= u_1^{\hat{x}_1 + c\,\hat{y}_1 - z_1} u_2^{\hat{x}_2 + c\,\hat{y}_2 - z_2} v^{-w},\end{aligned}$$

where $\hat{x}_1 = w\,x_1 \bmod q$, $\hat{x}_2 = w\,x_2 \bmod q$, $\hat{y}_1 = w\,y_1 \bmod q$ and $\hat{y}_2 = w\,y_2 \bmod q$. Each player computes a share of $\tilde{V}$. This idea was introduced in [4].

For this to be done, players generate a random factor w for each ciphertext and multiply them by private keys to get $\hat{x}_1 := w\,x_1 \bmod q$ and so on. This is where our multiplication protocol is applied. This step takes 2-moves but can be done before receiving the ciphertext. The decryption protocol can then be performed non-interactively.

Key Generation: Players generate a secret key $(x_1, x_2, y_1, y_2, z_1, z_2)$ by using *RandVSS* as follows.

$$\begin{aligned}\mathit{RandVSS}([x_1],[x_2])[g_1,g_2] &\rightarrow (x_{1i}, x_{2i})(X, EX_1, \ldots, EX_t) \\ \mathit{RandVSS}([y_1],[y_2])[g_1,g_2] &\rightarrow (y_{1i}, y_{2i})(Y, EY_1, \ldots, EY_t) \\ \mathit{RandVSS}([z_1],[z_2])[g_1,g_2] &\rightarrow (z_{1i}, z_{2i})(Z, EZ_1, \ldots, EZ_t)\end{aligned}$$

Each player computes verification commitments $VX_i = X \prod_{k=1}^{t} {EX_k}^{i^k}$ and $VY_i = Y \prod_{k=1}^{t} {EY_k}^{i^k}$ for all $i \in \mathcal{P}$, and checks if they do not equal 1. Each player i then shares (z_{1i}, z_{2i}) as

$$\mathit{PedVSS}(z_{1i}, z_{2i})[g_1,g_2] \xrightarrow{f_1, f_2} (z_{1ij}, z_{2ij})(\langle VZ_i \rangle, EZ_{i1}, \ldots, EZ_{it})$$

to prepare for decryption where VZ_i be verification commitments for shares z_{1i}, z_{2i} such that $VZ_i = \prod_{k=0}^{t} {EZ_k}^{i^k}$.

Key Randomization: Before performing the decryption protocol, players need to generate a shared random number w and obtain shares of products wx_1, wx_2, wy_1, and wy_2. Intuitively, this process corresponds to randomizing the private keys. In the following, g_3 is a random element of G_q whose indices for g_1 and g_2 are unknown.

[**Protocol :** KEY RANDOMIZATION]

KR-1 The players perform

$$RandVSS([w],[w'])[g_1,g_2] \rightarrow (w_i, w'_i)(EW_0,\ldots,EW_t).$$

Let VW_i denote the verification commitment for w_i and w'_i.

KR-2 Each player P_i performs the following VSS-s with t-degree random polynomials $f_3,\ldots,f_7$, $d_1,\ldots,d_5$ whose free terms are chosen properly as indicated in left parentheses. $\gamma_{1i},\ldots,\gamma_{4i}$ are randomly chosen from Z_q.

$$\begin{aligned}
PedVSS(w_i, w'_i)[g_1,g_2] &\xrightarrow{f_3,d_1} (w_{ij}, w'_{ij})(\langle VW_i\rangle, EW_{i1},\cdots)\\
PedVSS(w_i, \gamma_{1i})[VX_i, g_3] &\xrightarrow{f_3,d_2} (\langle w_{ij}\rangle, \gamma_{1ij})(EWX_{i0},\cdots)\\
PedVSS(w_i x_{1i}, w_i x_{2i}, \gamma_{2i})[g_1,g_2,g_3] &\xrightarrow{f_4,f_5,d_3} (\tilde{x}_{1ij}, \tilde{x}_{2ij}, \gamma_{2ij})\\
&\qquad (\langle EWX_{i0}\rangle, E\tilde{X}_{i1},\cdots)\\
PedVSS(w_i, \gamma_{3i})[VY_i, g_3] &\xrightarrow{f_3,d_4} (\langle w_{ij}\rangle, \gamma_{3ij})(EWY_{i0},\cdots)\\
PedVSS(w_i y_{1i}, w_i y_{2i}, \gamma_{4i})[g_1,g_2,g_3] &\xrightarrow{f_6,f_7,d_5} (\tilde{y}_{1ij}, \tilde{y}_{2ij}, \gamma_{4ij})\\
&\qquad (\langle EWY_{i0}\rangle, E\tilde{Y}_{i1},\cdots)
\end{aligned}$$

KR-3 Each player j verifies everything in a way similar to that used in **DM-2**. Player j then computes its share of the randomized private keys as

$$(\hat{x}_{1j},\hat{x}_{2j},\hat{\gamma}_{2j}) := \left(\sum_{i\in\mathcal{I}}\lambda_{i,\mathcal{I}}\tilde{x}_{1ij},\ \sum_{i\in\mathcal{I}}\lambda_{i,\mathcal{I}}\tilde{x}_{2ij},\ \sum_{i\in\mathcal{I}}\lambda_{i,\mathcal{I}}\gamma_{2ij}\right), \text{ and}$$

$$(\hat{y}_{1j},\hat{y}_{2j},\hat{\gamma}_{4j}) := \left(\sum_{i\in\mathcal{I}}\lambda_{i,\mathcal{I}}\tilde{y}_{1ij},\ \sum_{i\in\mathcal{I}}\lambda_{i,\mathcal{I}}\tilde{y}_{2ij},\ \sum_{i\in\mathcal{I}}\lambda_{i,\mathcal{I}}\gamma_{4ij}\right)$$

in modulo q, where $\mathcal{I}$ is a set of qualified players. Player j also computes verification commitments

$$V\hat{X}_i := \prod_{k=0}^{t}\left(\prod_{\ell\in\mathcal{I}} E\tilde{X}_{\ell k}^{\lambda_{\ell,\mathcal{I}}}\right)^{i^k}, \text{and}$$

$$V\hat{Y}_i := \prod_{k=0}^{t}\left(\prod_{\ell\in\mathcal{I}} E\tilde{Y}_{\ell k}^{\lambda_{\ell,\mathcal{I}}}\right)^{i^k}$$

for all $i\in\mathcal{I}$ to prepare for subsequent use. (Note that $V\hat{X}_i = g_1^{\hat{x}_{1i}} g_2^{\hat{x}_{2i}} g_3^{\gamma_{2i}}$ and $V\hat{Y}_i = g_1^{\hat{y}_{1i}} g_2^{\hat{y}_{2i}} g_3^{\gamma_{4i}}$.) Player j stores the shares and verification commitments.

[**End**]

Decryption:
[**Protocol :** Robust Threshold Decryption]

RD-1 Each player i broadcasts

$$\tilde{V_i} := u_1^{\hat{x}_{1i}+c\,\hat{y}_{1i}-z_{1i}} u_2^{\hat{x}_{2i}+c\,\hat{y}_{2i}-z_{2i}} v^{-w_i}$$

where $c := Hash(u_1, u_2, m) \bmod q$. Player i then performs the following four VSS-s. Here, t-degree polynomials $f_8, \ldots, f_{11}$ and d_6, d_7 are randomly chosen so that they have proper free terms as indicated in the parentheses in the right side.

$$PedVSS(\hat{x}_{1i}, \hat{x}_{2i}, \hat{\gamma}_{2i})[g_1, g_2, g_3] \overset{f_8, f_9, d_6}{\longrightarrow} (\hat{x}_{1ij}, \hat{x}_{2ij}, \hat{\gamma}_{2ij})(\langle V\hat{X}_i\rangle, \ldots)$$
$$PedVSS(\hat{y}_{1i}, \hat{y}_{2i}, \hat{\gamma}_{4i})[g_1, g_2, g_3] \overset{f_{10}, f_{11}, d_7}{\longrightarrow} (\hat{y}_{1ij}, \hat{y}_{2ij}, \hat{\gamma}_{4ij})(\langle V\hat{Y}_i\rangle, \ldots)$$

and

$$PedVSS(\hat{x}_{1i} + c\,\hat{y}_{1i} - z_{1i},\ \hat{x}_{2i} + c\,\hat{y}_{2i} - z_{2i},\ -w_i)[u_1, u_2, v] \overset{f_{12}, f_{13}, f_{14}}{\longrightarrow}$$
$$(\langle \hat{x}_{1ij} + c\,\hat{y}_{1ij} - z_{1ij}\rangle, \langle \hat{x}_{2ij} + c\,\hat{y}_{2ij} - z_{2ij}\rangle, \langle -w_{ij}\rangle)(\langle \tilde{V}_i\rangle, E\,\tilde{V}_{i1}, \cdots, E\,\tilde{V}_{it}),$$

where $f_{12} = f_8 + c\,f_{10} - f_1$, $f_{13} = f_9 + c\,f_{11} - f_2$, and $f_{14} = -f_3$.

RD-2 Each player j verifies

$$g_1^{\hat{x}_{1ij}}\, g_2^{\hat{x}_{2ij}}\, g_3^{\hat{\gamma}_{2ij}} = V\hat{X}_i \prod_{k=1}^{t} E\hat{X}_{ik}^{\ j^k},$$
$$g_1^{\hat{y}_{1ij}}\, g_2^{\hat{y}_{2ij}}\, g_3^{\hat{\gamma}_{4ij}} = V\hat{Y}_i \prod_{k=1}^{t} E\hat{Y}_{ik}^{\ j^k}, \text{ and}$$
$$u_1^{\hat{x}_{1ij}+c\,\hat{y}_{1ij}-z_{1ij}}\, u_2^{\hat{x}_{2ij}+c\,\hat{y}_{2ij}-z_{2ij}}\, v^{-w_{ij}} = \tilde{V}_i \prod_{k=1}^{t} E\tilde{V}_{ik}^{\ j^k}$$

for all i received. Disqualification will be done in the similar way as before. Player j finally obtains plaintext as

$$M = m / \prod_{i \in \mathcal{Q}} \tilde{V}_i^{\lambda_{i,\mathcal{Q}}}$$

where $\mathcal{Q}$ is a set of more than $t+1$ qualified players.

[End]

6.3 Security Analysis

Our definition of security against an adaptive chosen ciphertext attack refers to [9]. Similarly, a precise security model for the multi-party setting can be seen in [25]. In that model, there are adversaries inside the decryption oracle

who may leak internal information to an attacker. However, because individual private keys and verification result, i.e., V are information theoretically secure during the protocols, the joint view of the attacker and the insiders does not differ from that of the original scheme except for the following:

- Since *RandVSS* produces $w \in_U Z_q$, instead of Z_q^*, $w = 0$ happens with probability $1/q$. In this case, $\tilde{V} = 1$ holds regardless of V. Accordingly, an incorrect message is accepted with probability $1/q$.
- An adversary has a chance to get $\tilde{V}$ to be 1 for incorrect messages by adjusting corrupted player's share $\tilde{V}_i$ after seeing shares sent from uncorrupted players. However, this forces the corrupt player i to share $\tilde{V}_i$ without knowing its witnesses in **RD-1**. Hence the success probability is $1/q$.
- Honest player j can be disqualified with probability $2/q$, because of $VX_j = 1$ or $VY_j = 1$.

All of those increases in the probability of accepting incorrect messages are negligible in $|q|$ if the number of adversaries is limited to polynomial in $|q|$. Thus, the players reject all invalid ciphertexts except with negligible probability. The rest of the protocol is the same as the proof of security for the original scheme.

7 Conclusion

We have shown a non-interactive distributed multiplication protocol in standard cryptographic setting with a private network and broadcast channels. Non-interactive means that players need to use the private network and broadcast channel only once to send data without the need to synchronize with other players. The protocol withstands less than $n/2$ corrupt players and uses no zero-knowledge interactive proofs. Compared to the previous four-move scheme that relies on zero-knowledge proofs, it increases computation and communication complexity according to the factor of threshold t.

As an application of our protocol, we constructed a threshold version of the Cramer-Shoup cryptosystem which works non-interactively with precomputation for randomizing private keys. If the key randomization protocol is combined to the decryption protocol, it requires four data moves. Although this form still enjoys less round complexity than that of the previous construction (including precomputation steps), it raises an open problem: Show an optimally resilient and non-interactive protocol that generates a random number and multiplies it by a previously shared secret.

Acknowledgments

The author wishes to thank Rosario Gennaro for his invaluable suggestions. Comments from Tatsuaki Okamoto and Eiichiro Fujisaki on an early version of this paper are also appreciated.

References

1. D. Beaver. Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority. *Journal of Cryptology*, 4:75–122, 1991.
2. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cyptographic fault-tolerant distributed computation. In *Proceedings of the* 20*th annual ACM Symposium on the Theory of Computing*, pages 1–10, 1988.
3. D. Boneh and M. Franklin. Efficient generation of shared RSA keys. In B. S. Kaliski Jr., editor, *Advances in Cryptology — CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 425–439. Springer-Verlag, 1997.
4. R. Canetti and S. Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 90–106. Springer-Verlag, 1999.
5. M. Cerecedo, T. Matsumoto, and H. Imai. Efficient and secure multiparty generation of digital signatures based on discrete logarithms. *IEICE Transaction of Fundamentals of electronic Communications and Computer Science*, E76-A(4):532–545, April 1993.
6. D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *Proceedings of the* 20*th annual ACM Symposium on the Theory of Computing*, pages 11–19, 1988.
7. R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations secure against an adaptive adversary. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 311–326. Springer-Verlag, 1999.
8. R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In W. Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 103–118. Springer-Verlag, 1997.
9. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer-Verlag, 1998.
10. Y. G. Desmedt and Y. Frankel. Threshold cryptosystems. In G. Brassard, editor, *Advances in Cryptology — CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 307–315. Springer-Verlag, 1990.
11. P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proceedings of the* 28*th IEEE Annual Symposium on Foundations of Computer Science*, pages 427–437, 1987.
12. P. Feldman and S. Micali. Optimal algorithms for byzantine agreement. In *Proceedings of the 20th ACM Symposium on the Theory of Computing*, pages 148–161, 1988.
13. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology — CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–199. Springer-Verlag, 1986.
14. Y. Frankel, P. D. MacKenzie, and M. Yung. Robust efficient distributed RSA-key generation. In *Proceedings of the* 30*th annual ACM Symposium on Theory of Computing*, pages 663–672, New York City, 1998.

15. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. In U. Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 354–371. Springer-Verlag, 1996.
16. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 295–310. Springer-Verlag, 1999.
17. R. Gennaro, M. Rabin, and T. Rabin. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In *17th ACM Symposium on Principles of Distributed Computing*, 1998.
18. O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *Proceedings of the 27th IEEE Annual Symposium on Foundations of Computer Science*, pages 174–187, 1986.
19. A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive secret sharing or: How to cope with perpetual leakage. In D. Coppersmith, editor, *Advances in Cryptology — CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 339–352. Springer-Verlag, 1995.
20. M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 2(27):228–234, 1980.
21. T. P. Pedersen. A threshold cryptosystem without a trusted party. In D. W. Davies, editor, *Advances in Cryptology — EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, pages 522–526. Springer-Verlag, 1991.
22. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer-Verlag, 1992.
23. T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the 21st annual ACM Symposium on the Theory of Computing*, pages 73–85, 1989.
24. A. Shamir. How to share a secret. *Communications of the ACM*, 22:612–613, 1979.
25. V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. In K. Nyberg, editor, *Advances in Cryptology — EUROCRYPT '98*, Lecture Notes in Computer Science, pages 1–16. Springer-Verlag, 1998.