A Dynamic Tradeoff between Active and Passive Corruptions in Secure Multi-Party Computation

Martin Hirt[1], Ueli Maurer[1], and Christoph Lucas[2,⋆]

[1] ETH Zurich
{hirt,maurer}@inf.ethz.ch
[2] ETH Zurich and Ergon Informatik AG
christoph.lucas@ergon.ch

Abstract. At STOC '87, Goldreich et al. presented two protocols for secure multi-party computation (MPC) among n parties: The first protocol provides *passive* security against $t < n$ corrupted parties. The second protocol provides even *active* security, but only against $t < n/2$ corrupted parties. Although these protocols provide security against the provably highest possible number of corruptions, each of them has its limitation: The first protocol is rendered completely insecure in presence of a single active corruption, and the second protocol is rendered completely insecure in presence of $\lceil n/2 \rceil$ passive corruptions.

At Crypto 2006, Ishai et al. combined these two protocols into a single protocol which provides passive security against $t < n$ corruptions and active security against $t < n/2$ corruptions. This protocol unifies the security guarantees of the passive world and the active world ("best of both worlds"). However, the corruption threshold $t < n$ can be tolerated only when *all* corruptions are passive. With a single active corruption, the threshold is reduced to $t < n/2$.

As our main result, we introduce a *dynamic tradeoff* between active and passive corruptions: We present a protocol which provides security against $t < n$ passive corruptions, against $t < n/2$ active corruptions, *and everything in between.* In particular, our protocol provides full security against k active corruptions, as long as less than $n - k$ parties are corrupted in total, for any unknown k.

The main technical contribution is a new secret sharing scheme that, in the reconstruction phase, releases secrecy *gradually.* This allows to construct non-robust MPC protocols which, in case of an abort, still provide some level of secrecy. Furthermore, using similar techniques, we also construct protocols for reactive MPC with hybrid security, i.e., different thresholds for secrecy, correctness, robustness, and fairness. Intuitively, the more corrupted parties, the less security is guaranteed.

Keywords: Multi-party computation, gradual secret sharing, computational security, mixed adversary.

⋆ Work done while the author was at ETH Zurich.

R. Canetti and J.A. Garay (Eds.): CRYPTO 2013, Part II, LNCS 8043, pp. 203–219, 2013.

1 Introduction

1.1 Secure Multi-Party Computation

Multi-Party Computation (MPC) allows a set of n parties to securely compute a (probabilistic) function f in a distributed manner, where security means that secrecy of the inputs and correctness of the output are maintained even when some of the parties are dishonest. The dishonesty of parties is modeled with a central adversary who corrupts parties. The adversary can be *passive*, i.e., can read the internal state of the corrupted parties, or *active*, i.e., can make the corrupted parties deviate arbitrarily from the protocol. Reactive MPC considers the more general case where parties can provide inputs even after having received (intermediate) outputs.

MPC was originally proposed by Yao [Yao82]. The first general solution was provided in [GMW87], where two protocols are presented, one providing passive security against any number of corruptions, and one providing active security against a faulty minority. These protocols are computationally secure only. Information-theoretically secure MPC was considered in [BGW88, CCD88, RB89, Bea91].

1.2 Extensions of the Basic Setting

These seminal MPC results have been generalized and extended in numerous directions, among which we focus on those most relevant for us: The strict separation between active and passive adversaries was overcome in [Cha89, DDWY93, FHM98, HMZ08] by considering an adversary that corrupts some parties actively and some additional parties passively. Such a *mixed adversary* is characterized by two fixed thresholds, indicating the maximum number of actively and passively corrupted parties, respectively.

A more fine-grained analysis of the achieved security guarantees was initiated in [Cha89] and further advanced in [FHHW03, FHW04, Kat07, LRM10, HLMR11, HLMR12]: These protocols provide *hybrid security*, i.e., depending on the actual adversary, only a subset of the usual security guarantees (secrecy, correctness, robustness, fairness) are guaranteed. Intuitively, the more parties are corrupted, the less security is guaranteed.

For completeness, the considered models and achieved security levels of the mentioned protocols are summarized in Appendix A.

1.3 Prior Work

In their seminal paper [GMW87], the authors provide two different protocols, one for passive security against up to $t < n$ corruptions, and one for active security against up to $t < \frac{n}{2}$ corruptions. In [IKLP06], these two protocols are combined into a single protocol, which is secure against an adversary that either

passively corrupts any number of parties or actively corrupts a minority of the parties. This combined protocol is only applicable for non-reactive functions, and it is proven that this combination is impossible for reactive MPC. For this more general setting, the authors present a protocol in the active world that provides full security up to a first threshold t, and correctness and secrecy up to a second threshold s, given that $t < n/2$ and $s + t < n$.

Note that all provided protocols are secure against an adversary that is either fully passive or fully active. In particular, the protocol for non-reactive MPC is rendered completely insecure when the adversary corrupts $\lceil n/2 \rceil$ parties, even if all but one corruptions are only passive.

1.4 Contributions

We present an MPC protocol (for non-reactive functions) with a dynamic tradeoff between active and passive corruptions. As [IKLP06], the protocol provides the best possible security level in presence of a purely passive adversary (namely $t < n$) as well as in presence of a purely active adversary (namely $t < n/2$). In addition, the protocol also tolerates mixed adversaries that corrupt some parties actively and some other parties passively, as long as at most k parties are corrupted actively and at most $n - k - 1$ parties are corrupted in total. Note that k need *not* be known, as it is not a parameter of the protocol.

In order to construct such a protocol, we introduce the notion of *gradual* verifiable secret sharing (VSS). In contrast to traditional VSS, a gradual VSS reduces the number of adversaries against which secrecy is guaranteed during reconstruction in a step-wise fashion, and at the same time increases the number of adversaries against which robustness is guaranteed. By that, if the reconstruction of a secret aborts, secrecy against many adversaries is still guaranteed.

Furthermore, we generalize and extend our results in two directions: First, we work in a model with hybrid security. That means, we consider each security guarantee (correctness, secrecy, robustness, and fairness) separately, and our protocols provide each guarantee against as many corrupted parties as possible. Second, in the setting of reactive MPC, we extend the protocol from [IKLP06] with fairness and security against mixed adversaries, while at the same time removing the restriction that robustness can only be guaranteed against a corrupted minority.

1.5 Outline of the Paper

The paper is organized as follows: The model used in this work is described in Section 2. In Section 3, we briefly review the standard definition of VSS and introduce the notion of gradual reconstruction. Furthermore, we provide gradual VSS protocols for threshold adversaries. In Sections 4 and 5, we present protocols for non-reactive and reactive MPC, respectively, together with optimal bounds.

2 Model

2.1 Parties

We consider n parties $p_1, \ldots, p_n$, connected by pairwise synchronous secure channels and authenticated broadcast channels,[1] who want to implement an ideal functionality $\mathcal{F}$ computing a (probabilistic) function f over a finite field $\mathbb{F}$ with $|\mathbb{F}| > n$. Without loss of generality, we assume only public outputs (possibly several at the same time). Local outputs towards a designated party can be blinded with a random input from that party. For reactive MPC, $\mathcal{F}$ is not restricted to functions and can provide outputs before taking some other inputs. There is a central adversary with polynomially bounded computing power who corrupts some parties passively (and reads their internal state) or even actively (and makes them misbehave arbitrarily). We denote the set of actually actively (passively) corrupted parties by $\mathcal{D}^*$ ($\mathcal{E}^*$), where for ease of notation, we assume that $\mathcal{D}^* \subseteq \mathcal{E}^*$. Uncorrupted parties are called *honest*, non-actively corrupted parties are called *correct*. For ease of notation, we assume that if a party does not receive an expected message (or receives an invalid message), a default message is used instead.

2.2 Security

The security of our protocols is computational, i.e., based on some computational assumption. We say a security guarantee holds *computationally* if it holds against a computationally bounded adversary. We consider the five standard security guarantees: *Secrecy* means that the adversary learns nothing about the honest parties' inputs and outputs (except, of course, for what can be derived from the corrupted parties' inputs and outputs). *Correctness* means that all parties either output the right value or no value at all. *Robustness* means that the adversary cannot prevent the honest parties from learning their respective outputs. This last requirement turns out to be very demanding. Therefore, relaxations of full security have been proposed, where robustness is replaced by weaker output guarantees: *Fairness* means that the adversary can possibly prevent the honest parties from learning their outputs, but then also the corrupted parties do not learn their outputs. In the case of reactive MPC, fairness can only be achieved for outputs provided at the same time, i.e., for each output round, either all (honest) parties learn the outputs or also the adversary does not learn them. However, the adversary can abort the protocol after having received outputs from prior rounds. *Agreement on abort* means that the adversary can possibly prevent honest parties from learning their output, even while corrupted parties learn their outputs, but then the honest parties at least reach agreement on this

[1] Secure bilateral channels are usually established via standard techniques such as encryption and digital signatures. Broadcast channels are usually simulated with an appropriate protocol [DS82].

fact (and typically make no output).[2] The level of security (secrecy, correctness, fairness, robustness, agreement on abort) depends on $(\mathcal{D}^*, \mathcal{E}^*)$.

2.3 Characterization of Tolerated Adversaries

Traditionally, protocols for threshold adversaries are characterized by a single threshold t that specifies the maximal adversary that can be tolerated. This basic representation has been extended as follows: On the one hand, a mixed adversary is characterized by two thresholds (t_a, t_p), where he may corrupt up to t_p parties passively, and up to t_a of these parties even actively. To model security guarantees against incomparable maximal adversaries, we need to consider multiple pairs of thresholds. Therefore, we use multi-thresholds $T = \{(t_{a,1}, t_{p,1}), \ldots, (t_{a,k}, t_{p,k})\}$, i.e., sets of pairs of thresholds (t_a, t_p). In this model, security is guaranteed if $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq (t_a, t_p)$ for some $(t_a, t_p) \in T$, denoted by $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T$, where $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq (t_a, t_p)$ is a shorthand for $|\mathcal{D}^*| \leq t_a$ and $|\mathcal{E}^*| \leq t_p$. On the other hand, the level of security (correctness, secrecy, robustness, and fairness) depends on the number $(|\mathcal{D}^*|, |\mathcal{E}^*|)$ of *actually* corrupted parties (hybrid security). Hence, we consider the four multi-thresholds T^c, T^s, T^r, and T^f: Correctness with agreement on abort is guaranteed for $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^c$, secrecy is guaranteed for $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^s$, robustness is guaranteed for $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^r$, and fairness is guaranteed for $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^f$. We have the assumption that $T^r \leq T^c$ and $T^f \leq T^s \leq T^c$,[3] as secrecy and robustness are not well defined without correctness, and as fairness cannot be achieved without secrecy.

2.4 Ideal Functionality

For ease of presentation, we provide our proof sketches in a property-based security model. This allows to describe our ideas in a straightforward and understandable way. All statements could be made formal in a simulation-based model. To avoid ambiguity, we sketch the ideal functionality of our non-reactive MPC protocol in Figure 1.

We stress that in a setting without secrecy (i.e., $(\mathcal{D}^*, \mathcal{E}^*) \not\leq T^s$) the adversary may learn the inputs from honest parties before he has to provide inputs from the corrupted parties. Furthermore, for probabilistic functions, the adversary can freely choose the random string.

3 Gradual Verifiable Secret Sharing

We first briefly review the standard definition of verifiable secret sharing (VSS) schemes. Then, we define a new property for VSS schemes introducing the notion

[2] In our constructions, all abort decisions are based on publicly known values. Hence, we have agreement on abort for free. Note that the impossibility proofs hold even when agreement on abort is not required.

[3] We write $T_1 \leq T_2$ if $\forall (t_a, t_p) \in T_1 : (t_a, t_p) \leq T_2$.

Ideal Functionality $\mathcal{F}$: Given (T^r, T^f, T^s, T^c) and $(\mathcal{D}^*, \mathcal{E}^*) \leq T^c$.
1. Receive inputs from honest parties $\mathcal{P} \setminus \mathcal{E}^*$.
2. If $(\mathcal{D}^*, \mathcal{E}^*) \not\leq T^s$: send these inputs to the adversary.
3. Receive inputs from the adversary for the parties in $\mathcal{E}^*$.
4. For probabilistic functions only:
   If $(\mathcal{D}^*, \mathcal{E}^*) \leq T^s$: sample a random bit string r of appropriate length.
   Otherwise: Receive r from the adversary.
5. Evaluate the function.
6. If $(\mathcal{D}^*, \mathcal{E}^*) \not\leq T^f$: send the output to the adversary.
7. If $(\mathcal{D}^*, \mathcal{E}^*) \not\leq T^r$: receive a bit from the adversary, and abort if the bit is 1.
8. Send the output to the honest parties $\mathcal{P} \setminus \mathcal{E}^*$ and to the adversary.

Fig. 1. Sketch of the Ideal Functionality for non-reactive MPC

of gradual reconstruction.[4] Finally, we present schemes that achieve the new requirements.

3.1 Definitions

A *Verifiable Secret Sharing* (VSS) scheme allows a designated party (the *dealer*) to share a value s among all parties, such that the parties can jointly reconstruct the value. The following definition captures the standard, well-known properties of verifiable secret sharing:

Definition 1 (VSS). *A (T^s, T^r)-secure* Verifiable Secret Sharing *(VSS) is a pair of protocols* `Share` *and* `Rec`*, where* `Share` *takes input s from the dealer and* `Rec` *gives output s' to each party, if the following conditions are fulfilled:*
Secrecy: *If $(\mathcal{D}^*, \mathcal{E}^*) \leq T^s$, then in* `Share` *the adversary obtains no information about s.*
Correctness: *After* `Share`*, the dealer is bound to a value s', where $s' = s$ if the dealer is correct. Furthermore, in* `Rec`*, either each (correct) party outputs s' or all (correct) parties abort.*
Robustness: *The adversary cannot abort* `Share`*. If $(\mathcal{D}^*, \mathcal{E}^*) \leq T^r$, then the adversary cannot abort* `Rec`*.*

For $(\mathcal{D}^*, \mathcal{E}^*) \not\leq T^r$, this definition does not rule out that the reconstruction protocol aborts even in an unfair way, where the honest parties do not learn the secret but the corrupted parties do. In fact, most VSS schemes in the literature show this undesired behavior: When corrupted parties do not broadcast their shares, they still learn the shares from the honest parties and can compute the secret, but the honest parties do not obtain enough shares and abort.

Clearly, a certain level of unfairness cannot be avoided when secrecy and robustness are to be guaranteed with respect to many corruptions. In particular,

[4] This notion should not be confused with the notion of gradual release of secrecy as introduced by [Blu83].

whenever a sharing scheme is secret with respect to some subset $M \subseteq \mathcal{P}$ of the parties, then it cannot be robust with respect to the complement $\mathcal{P} \setminus M$ of this subset: When the parties in M have no information about the shared value after `Share`, and the parties in $\mathcal{P} \setminus M$ do not participate in `Rec`, then the value cannot be reconstructed. Hence, the collection of subsets against which a sharing is secret implicitly defines the collection of subsets that can abort reconstruction (namely, the complements). In usual reconstruction protocols, all correct parties directly broadcast their entire shares, i.e., secrecy is given up against all subsets at once, before robustness against a single subset is achieved. This means that during reconstruction, any subset of parties that can abort, can also abort in an unfair way. Our new definition below requires that the transition from secrecy to robustness is gradual, such that when a small set of parties does not broadcast their share, then only a large subset of parties jointly obtains information about the secret.

Definition 2 (Gradual VSS). *A (T^s, T^r)-secure VSS is* gradual *if the following conditions are fulfilled: If* `Rec` *aborts, each party outputs a non-empty set $B \subseteq \mathcal{D}^*$, and the adversary obtained no information about the secret s if $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^s$ and $|\mathcal{E}^*| < n - |B|$.*

3.2 A Gradual VSS Scheme

We describe a gradual VSS scheme based on the standard Shamir sharing scheme [Sha79], and extended with (homomorphic) commitments to provide verifiability (e.g. [Ped91]). To obtain the gradual property, summands $s_1, \ldots, s_d$ with $s_1 + \ldots + s_d = s$ are chosen at random and, rather than the secret itself, these summands are shared, where summand s_i is shared with degree i. Then, during reconstruction, the summands are reconstructed one by one, in decreasing order of the sharing degree. We assume that each party p_i is assigned a unique and publicly known evaluation point $\alpha_i \in \mathbb{F} \setminus \{0\}$,[5] and that the commitments are homomorphic and transferable by sending the opening information. This construction results in the scheme $\texttt{VSS}^d = (\texttt{Share}^d, \texttt{Rec}^d)$ for parameter d.

Definition 3 (d-sharing). *A value s is* d-shared, *denoted by $[s]_d$, if there are values $s_1, \ldots, s_d$, such that $s_1 + \ldots + s_d = s$ and, for all $i \in \{1, \ldots, d\}$, there is a polynomial $g_i(x)$ of degree i with $g_i(0) = s_i$, and every party p_j holds a share $s_{ij} = g_i(\alpha_j)$ and is committed to it.*

The sharing protocol from [Ped91] can be extended in a straightforward way to compute such a d-sharing. A description of the protocol can be found in Figure 2. This share protocol provides resilience even against a corrupted dealer. It turns out that in our protocols, essentially only ideal functionalities need to compute d-sharings. Trivially, given a value s, such an honest dealer can directly sample and distribute a correct sharing $[s]_d$ without running $\texttt{Share}^d$. The (probabilistic) function that samples shares of some given input s is denoted by $\texttt{State}^d$.

[5] This implies that the field $\mathbb{F}$ must have more than n elements.

Protocol $\texttt{Share}^d$: Given input s from the dealer, compute a d-sharing of this value.

1. The dealer chooses uniformly random summands $s_1, \ldots, s_d$ with $\sum_{i=1}^{d} s_i = s$.
2. For $i \in \{1, \ldots, d\}$:
   (a) The dealer chooses a random polynomial $g_i(x)$ of degree i with $g_i(0) = s_i$, and computes and broadcasts (homomorphic) commitments of the coefficients of $g_i(x)$.
   (b) For each share $s_{ij} = g_i(\alpha_j)$, each party locally computes a commitment c_{ij} (using the homomorphic property), and the dealer sends the corresponding opening information o_{ij} to party p_j. Then, p_j broadcasts a complaint bit, indicating whether o_{ij} opens c_{ij} to some value s'_{ij}.
   (c) For each share s_{ij} for which an inconsistency was reported, the dealer broadcasts the opening information o_{ij}, and if o_{ij} opens c_{ij}, p_j accepts o_{ij}. Otherwise, the dealer is disqualified (and a default sharing of a default value is used).
3. Each party p_j outputs its share $\big((s_{1j}, o_{1j}), \ldots, (s_{dj}, o_{dj})\big)$ and all commitments.

Fig. 2. The share protocol for threshold adversaries

Lemma 1. *Given a parameter $d < n$ and input s, $\texttt{Share}^d$ robustly computes a correct d-sharing $[s]_d$. If $|\mathcal{E}^*| \leq d$, the adversary obtains no information about s.*

Proof. Correctness: Trivially, in Step 2.a, any (well-formed) commitments broadcasted by the dealer are correct. In Step 2.b, commitments to all shares are computed locally by each party directly from the commitments to the coefficients broadcasted in Step 2.a. Hence, all (correct) parties have a consistent view with correct commitments. In Steps 2.b and 2.c, due to the binding property of the commitments, the adversary cannot distribute inconsistent opening information without being detected. Hence, the sharing is correct (or the dealer is disqualified and a default sharing is used).
Secrecy: The commitments are computationally hiding. Therefore, the adversary obtains no information in Step 2.a of $\texttt{Share}^d$. Furthermore, the summand s_d is shared with degree d. Hence, in Step 2.b, if not more than d parties are passively corrupted, the adversary obtains no information about s_d, and therefore not about s. In Step 2.c, whenever a value is broadcasted, the adversary knew this value already beforehand.
Robustness: By inspection, the share protocol does not abort.

In Figure 3, we describe the reconstruction protocol for a single sharing. Clearly, this protocol can be extended to reconstruct multiple sharings in parallel by executing the protocol on a vector of sharings, where an abort in one instance implies an immediate abort (in the same round) for all instances.

Lemma 2. *Given is a d-sharing $[s]_d$ for $d < n$. If $|\mathcal{D}^*| < n - d$, then $\texttt{Rec}^d$ (robustly) outputs s to all parties. Otherwise, either it outputs s to all parties, or it aborts and outputs a non-empty set $B \subseteq \mathcal{D}^*$, and the adversary obtained no information about the secret if $|\mathcal{E}^*| < n - |B|$.*

Protocol $\texttt{Rec}^d$: Given a d-sharing of some value s, reconstruct s to all parties.

1. For $i = d$ down to 1:
   (a) Each party p_j opens the commitment to its share s_{ij} via broadcast.
   (b) If at least $i+1$ parties correctly opened the commitments to their respective shares, each party locally interpolates $g_i(x)$ and computes $s_i = g_i(0)$. Otherwise, the protocol is aborted and each party outputs the set B of parties that did not broadcast correct opening information.
2. Each party outputs $s = s_1 + \ldots + s_d$.

Fig. 3. The protocol for gradual reconstruction for threshold adversaries

Proof. CORRECTNESS: The only operation in the protocol is the opening of commitments. Hence, given a correct sharing and the binding property of the commitment scheme, incorrect parties cannot deviate without being detected.
ROBUSTNESS: To abort the reconstruction of some s_i, at least $n - i \geq n - d$ parties must refuse to correctly open their respective commitments. Hence, for $|\mathcal{D}^*| < n - d$, the protocol is robust.
GRADUAL: The reconstruction aborts (with B) only if in the ith iteration (for some i), the reconstruction of s_i failed. In that case, strictly less than $i+1$ parties opened their commitments correctly, hence $|B| \geq n - i$. Clearly, $B \subseteq \mathcal{D}^*$, since only active parties do not open their commitments correctly. Furthermore, if $|\mathcal{E}^*| < n - |B|$, the adversary has no information about $s_{|\mathcal{E}^*|}$ since the reconstruction of $s_{|\mathcal{E}^*|}$ did not yet start (note that $|\mathcal{E}^*| < n - |B| \leq i$).

The following corollary summarizes Lemma 1 ($\texttt{Share}^d$) and Lemma 2 ($\texttt{Rec}^d$):

Corollary 1. *Given a parameter $d < n$, $\texttt{VSS}^d = (\texttt{Share}^d, \texttt{Rec}^d)$ is a computationally (T^s, T^r)-secure, gradual VSS where $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^s$ if $|\mathcal{E}^*| \leq d$, and $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^r$ if $|\mathcal{D}^*| < n - d$.*

4 Non-reactive Multi-Party Computation

4.1 Overview

Our protocol for non-reactive MPC is based on an idea from [IKLP06]: Given the function f and the inputs $x_1, \ldots, x_n$, the protocol first distributedly computes $y = f(x_1, \ldots, x_n)$ using a correct and secret, but non-robust MPC protocol. Yet, instead of y itself, this MPC protocol outputs a sharing of y that was computed according to some VSS scheme. Then, the parties reconstruct this sharing.

In [IKLP06], whenever the non-robust MPC protocol aborts, the computation of $y = f(x_1, \ldots, x_n)$ is repeated with a robust MPC protocol, which provides security against an actively corrupted minority. Yet, if the reconstruction of y aborts, the adversary might already have learned the output, and repeating the computation would violate security. In contrast, by using a gradual VSS scheme to share y (cf. Section 3), our protocols achieve stronger security guarantees.

Given a set of actually (actively) cheating parties, a gradual VSS allows to maintain as much secrecy as possible. Then, in case of an abort during the reconstruction, the cheaters are identified and eliminated, and if the gradual VSS still guarantees enough secrecy,[6] the computation of $y = f(x_1, \ldots, x_n)$ is repeated using again the same MPC protocol among the remaining parties. Otherwise, the execution halts.

The protocol in [GMW87] (in the following denoted by GMW) provides security with abort for $t < n$ corrupted parties (i.e., correctness and secrecy against $t < n$ corrupted parties, and, in case of an abort, each correct party outputs the same non-empty set $B \subseteq \mathcal{D}^*$). However, it can easily be seen that correctness (but not secrecy) can also be achieved for $t = n$ corrupted parties.[7] We use GMW to implement the ideal functionality computing f and then a sharing of the result y.[8]

4.2 Construction

We use the gradual VSS scheme described in Section 3.2 with degree $d = n-1$. In fact, we only require the reconstruction protocol $\texttt{Rec}^{n-1}$ and the (probabilistic) function $\texttt{State}^{n-1}$ that, given a value y, samples shares of y according to $\texttt{VSS}^{n-1}$. Furthermore, the protocol receives a robustness parameter e stating the number of actively corrupted parties that the protocol can eliminate (and then repeat the run) without violating security. A set of parties is eliminated by removing the parties from $\mathcal{P}$ and reducing n and e accordingly. For details see Figure 4.

Lemma 3. *Given a function f and a robustness parameter e, the protocol for non-reactive MPC computes f in presence of an adversary corrupting $(|\mathcal{D}^*|, |\mathcal{E}^*|)$. It is always correct, robust if $|\mathcal{D}^*| \leq e$, secret if $|\mathcal{D}^*| + |\mathcal{E}^*| < n$ or $|\mathcal{E}^*| < n-e$, and fair if $|\mathcal{D}^*| + |\mathcal{E}^*| < n$.*

Proof. CORRECTNESS and ROBUSTNESS follow trivially by inspection.
SECRECY: GMW is secret for any number of corrupted parties. Furthermore, since $|\mathcal{E}^*| \leq n-1$ (otherwise there is no secrecy requirement), it follows from Corollary 1 that the output $[y]_{n-1}$ reveals no information to the adversary. Hence, he obtains no information about the inputs in Step 1. Steps 2 and 3 are independent from the inputs given the output. Therefore, if the protocol does not abort,

[6] The protocols are described with respect to a robustness parameter rather than a secrecy parameter as suggested here. It turns out that this simplifies the description and the proof.

[7] In particular this holds also in the setting with mixed adversaries where some parties are actively and all remaining parties are passively corrupted. This follows from the fact that each party has to prove the correctness of the messages it sends using a zero-knowledge protocol. Given instant randomness (i.e., randomness generated only when needed), even the challenges of passively corrupted parties are unpredictable to the adversary.

[8] Vanilla [GMW87] considers only Boolean circuits. However, any ideal functionality can be converted into a Boolean circuit in a straightforward way.

Non-reactive MPC: Given are a function f and a robustness parameter e.

1. Employ $\texttt{GMW}$ to first compute $y = f(x_1, \ldots, x_n)$, where x_i is the input from party p_i, then evaluate $\texttt{State}^{n-1}$ on y, and finally output to each party its corresponding share, resulting in $[y]_{n-1}$. If $\texttt{GMW}$ aborts with a set B of active cheaters, repeat with $\mathcal{P} = \mathcal{P} \setminus B$, $n = n - |B|$, and $e = e - |B|$.
2. Invoke $\texttt{Rec}^{n-1}$ on $[y]_{n-1}$. On abort with a set B of active cheaters: If $|B| \leq e$, then repeat the whole protocol with $\mathcal{P} = \mathcal{P} \setminus B$, $n = n - |B|$, and $e = e - |B|$. Otherwise, halt the execution.
3. Output y.

Fig. 4. The protocol for non-reactive MPC for threshold adversaries

secrecy is maintained. Yet, secrecy may be violated if the adversary can force a repetition of the protocol after learning the output.[9] If the protocol aborts in Step 2 with $B \subseteq \mathcal{D}^*$, then in the case $|\mathcal{D}^*| + |\mathcal{E}^*| < n$ we directly have $|\mathcal{E}^*| < n - |\mathcal{D}^*| \leq n - |B|$, hence secrecy is maintained. In the case $|\mathcal{E}^*| < n - e$, we either have that $|\mathcal{E}^*| < n - |B|$ (and secrecy is maintained), or $|\mathcal{E}^*| \geq n - |B|$, hence $|B| \geq n - |\mathcal{E}^*| > e$ and the protocol aborts, i.e. the adversary learns at most one output value.
Fairness is a subcase of secrecy and therefore omitted.

Given Lemma 3, we can derive a tight bound for (non-reactive) MPC:

Theorem 1. *In the model with broadcast and multi-threshold adversaries, computationally secure (non-reactive) MPC among n parties with thresholds T^c, T^s, T^r, and T^f, where $T^f \leq T^s \leq T^c$ and $T^r \leq T^c$, is possible if either $T^s = \{(0,0)\}$, or*

$$\Big(\forall (t_a^s, t_p^s) \leq T^s, (t_a^r, \cdot) \leq T^r : t_a^r + t_p^s < n \ \vee \ t_a^s + t_p^s < n\Big)$$
$$\wedge \quad \Big(\forall (t_a^f, t_p^f) \leq T^f : t_a^f + t_p^f < n\Big).$$

This bound is tight: If violated, there are (non-reactive) functionalities that cannot be securely computed.

Proof. The proof of necessity can be found in Section 4.3. To prove sufficiency, first consider the (trivial) case $T^s = \{(0,0)\}$. Then, every party simply broadcasts its inputs and computes the function on the broadcasted values (c.f. Section 2.4).

Otherwise, we use the protocol in Figure 4 with $e = \hat{t_a^r}$, where $\hat{t_a^r}$ is the maximal t_a^r value in T^r.
Correctness is always guaranteed, and Robustness follows directly from the choice of e.
Secrecy: Since $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^s$, we immediately have that $|\mathcal{D}^*| + |\mathcal{E}^*| < n \ \vee \ \forall (t_a^r, \cdot) \leq T^r : t_a^r + |\mathcal{E}^*| < n$. Then, it follows from the choice of e that $|\mathcal{D}^*| + |\mathcal{E}^*| < n \ \vee \ e + |\mathcal{E}^*| < n$.

[9] In that case, the adversary may learn two evaluations of f for different inputs.

FAIRNESS: Since $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^f$ and $\forall (t_a^f, t_p^f) \leq T^f : t_a^f + t_p^f < n$, we immediately have $|\mathcal{D}^*| + |\mathcal{E}^*| < n$.

Theorem 2. *There exists a cryptographically secure multi-party computation protocol among n parties for non-reactive functionalities which is fully secure against all adversaries $(\mathcal{D}, \mathcal{E})$ with $|\mathcal{D}| + |\mathcal{E}| < n$.*

Proof. Apply Theorem 1 with $T^c = T^s = T^r = T^f = \{(k, n-k-1) : \ k = 0, \ldots, \lfloor \frac{n-1}{2} \rfloor\}$.

4.3 Proof of Necessity for Non-reactive MPC

In this section, we prove that the bound in Theorem 1 is necessary, i.e. if violated, (non-reactive) MPC is impossible. The bound in Theorem 1 is violated if $T^s \neq \{(0,0)\}$ and $\Big(\exists (t_a^s, t_p^s) \leq T^s, (t_a^r, \cdot) \leq T^r : \ t_a^r + t_p^s \geq n \ \wedge \ t_a^s + t_p^s \geq n\Big) \ \vee \Big(\exists (t_a^f, t_p^f) \leq T^f : \ t_a^f + t_p^f \geq n\Big)$

Case: $\exists (t_a^s, t_p^s) \leq T^s, (t_a^r, \cdot) \leq T^r : \ t_a^r + t_p^s \geq n \ \wedge \ t_a^s + t_p^s \geq n$. For the sake of contradiction, assume that there is a protocol for this setting, and without loss of generality assume that $t_p^s > 0$ (there is such a t_p^s because $T^s \neq \{(0,0)\}$). For each $E \subseteq \mathcal{P}$, let ℓ_E denote the first round in the protocol in which the parties in E jointly can efficiently compute the output. Among all subsets $E \subseteq \mathcal{P}$ with $|E| = t_p^s$, let $\mathcal{E}$ denote the one with minimal ℓ_E, i.e., $\mathcal{E} \in \{E \subseteq \mathcal{P} : \ |E| = t_p^s \ \wedge \nexists E' \subseteq \mathcal{P} : |E'| = t_p^s \wedge \ell_{E'} < \ell_E\}$. Now consider an adversary actively corrupting some subset $\mathcal{D} \subseteq \mathcal{E}$ with $|\mathcal{D}| = n - t_p^s$, and let him abort all parties in $\mathcal{D}$ in round $\ell - 1$. By assumption, the remaining t_p^s parties $\mathcal{P} \backslash \mathcal{D}$ cannot compute the output (corresponding to the actual inputs). However, the protocol must not abort, as the actual adversary could be $(\mathcal{D}^*, \mathcal{E}^*) = (\mathcal{D}, \mathcal{D})$, for which robustness is guaranteed as $(|\mathcal{D}^*|, |\mathcal{E}^*|) = (n - t_p^s, n - t_p^s) \leq (t_a^r, t_a^r) \leq T^r$. Hence, the remaining parties must again take inputs and set default values for the inputs of parties in $\mathcal{D}$, but this violates secrecy if the actual adversary is $(\mathcal{D}^*, \mathcal{E}^*) = (\mathcal{D}, \mathcal{E})$ (note that $(|\mathcal{D}^*|, |\mathcal{E}^*|) = (n - t_p^s, t_p^s) \leq T^s$), who then learns the output of this run as well as the output of the next run with default input values for the parties in $\mathcal{D}$ (note that $|\mathcal{E}^*| \geq 1$).

As an example, consider the following (generalized OT-) functionality: Each party p_i inputs three bits: $a_0^{(i)}$, $a_1^{(i)}$, and $b^{(i)}$ (with default input $a_0^{(i)} = a_1^{(i)} = b^{(i)} = 0$). Let $d = b^{(1)} \oplus \ldots \oplus b^{(n)}$. The output is $y = (a_d^{(1)}, \ldots, a_d^{(n)})$. The adversary lets one actively corrupted party input $b = 1$, and all others $b = 0$. Then, with the attack described above, the adversary learns both $y_0 = (a_0^{(1)}, \ldots, a_0^{(n)})$ and $y_1 = (a_1^{(1)}, \ldots, a_1^{(n)})$, which clearly is a violation of secrecy.

Case: $\exists (t_a^f, t_p^f) \leq T^f : \ t_a^f + t_p^f \geq n$. Again, assume that there is a protocol for this setting, and let $\mathcal{E}$ denote the subset among all subsets $E \subseteq \mathcal{P}$ with $|E| = t_p^f$ such that ℓ_E is minimal (see Section 4.3). Consider the adversary

$(\mathcal{D}^*, \mathcal{E}^*) = (\mathcal{D}, \mathcal{E})$ for $\mathcal{D} \subseteq \mathcal{E}$ with $|\mathcal{D}| = n - t_p^f$, and let him abort all parties in $\mathcal{D}^*$ in round $\ell - 1$. By assumptions, the remaining t_p^f parties in $\mathcal{P} \setminus \mathcal{D}^*$ are not able to efficiently compute the output, whereas the adversary $(\mathcal{D}^*, \mathcal{E}^*)$ does learn the output, a violation of fairness.

5 Reactive Multi-Party Computation

5.1 Overview

For our protocol for reactive MPC, we adapt an idea from [IKLP06] and modify the given functionality $\mathcal{F}$ as follows: For each output y, instead of the value itself, it outputs a sharing of y that was computed according to some VSS scheme. Then, to obtain the output, the parties reconstruct this sharing. This modified $\mathcal{F}$ is implemented using an MPC protocol that is always correct, and as robust and secret as some (second) VSS scheme.

In contrast to [IKLP06], we use a gradual VSS scheme for the modification of $\mathcal{F}$. The gradual property allows to provide fairness beyond robustness. In fact, we only require the (probabilistic) function `State` that, given a value y, samples shares of y according to the gradual VSS scheme. We modify $\mathcal{F}$ such that it invokes `State` on each output value y, and then outputs the shares of y (instead of y itself). We modify $\mathcal{F}$ to use $\texttt{VSS}^d$ (Section 3.2) and denote the resulting functionality by $\mathcal{F}^d$.

The MPC protocol implementing the (modified) functionality $\mathcal{F}$ receives as parameter a (T^s, T^r)-secure VSS, and then provides correctness for any number of corrupted parties, secrecy if $(\mathcal{D}^*, \mathcal{E}^*) \leq T^s$, and robustness if $(\mathcal{D}^*, \mathcal{E}^*) \leq T^r$. Furthermore, if the protocol is aborted, then each party outputs the same non-empty set $B \subseteq \mathcal{D}^*$. Clearly, the non-robust protocol used in Section 4 can be extended accordingly with a VSS as described in [GMW87].[10] We instantiate the protocol using $\texttt{VSS}^d$ (Section 3.2) and denote the resulting protocol by $\texttt{GMW}^d$. Note that for this extension of `GMW`, a standard, non-gradual VSS would be sufficient.

5.2 Construction

We use the gradual VSS scheme described in Section 3.2 with the same sharing degree d for both the modification of $\mathcal{F}$, resulting in $\mathcal{F}^d$, and within `GMW`, resulting in $\texttt{GMW}^d$.

Lemma 4. *Given a functionality $\mathcal{F}$ and a parameter $d < n$, the protocol for reactive MPC implements $\mathcal{F}$ in presence of an adversary corrupting $(|\mathcal{D}^*|, |\mathcal{E}^*|)$. It is always correct, secret if $|\mathcal{E}^*| \leq d$, robust if $|\mathcal{D}^*| < n - d$, and fair if $|\mathcal{E}^*| \leq d \ \wedge \ |\mathcal{D}^*| + |\mathcal{E}^*| < n$.*

[10] The original description considers only VSS with a threshold of $n/2$. However, it is easy to see that any VSS can be used. The resulting protocol inherits the robustness and secrecy properties of the corresponding VSS, while leaving the correctness properties unchanged. The same holds for the simplified protocol in [Gol04, p. 735].

Reactive MPC: Given are a functionality $\mathcal{F}$ and a sharing degree d.

1. Invoke $\texttt{GMW}^d$ implementing $\mathcal{F}^d$.
2. On each output $[y]_d$, invoke $\texttt{Rec}^d$. If it aborts, halt the execution. Otherwise, output y.

Fig. 5. The protocol for reactive MPC for threshold adversaries

Proof. Correctness follows trivially by inspection, and Secrecy and Robustness follow immediately from Corollary 1. Fairness: Since $|\mathcal{E}^*| \leq d$, it follows from Corollary 1 that the adversary obtains no information in Step 1. Furthermore, if the reconstruction of an output value aborts with B, the gradual property guarantees that $B \subseteq \mathcal{D}^*$. Since $|\mathcal{D}^*| + |\mathcal{E}^*| < n$, we then have $|\mathcal{E}^*| < n - |\mathcal{D}^*| \leq n - |B|$, hence, the adversary did not obtain any information about y and fairness is preserved.

Given Lemma 4, we can derive a tight bound for reactive MPC:

Theorem 3. *In the model with broadcast and multi-threshold adversaries, computationally secure (reactive) MPC among n parties with thresholds T^c, T^s, T^r, and T^f, where $T^f \leq T^s \leq T^c$ and $T^r \leq T^c$, is possible if either $T^s = \{(0,0)\}$, or*

$$\forall (t_a^r, \cdot) \leq T^r, (\cdot, t_p^s) \leq T^s : t_a^r + t_p^s < n \quad \wedge \quad \forall (t_a^f, t_p^f) \leq T^f : t_a^f + t_p^f < n$$

This bound is tight: If violated, there are (reactive) functionalities that cannot be securely computed.

Proof. To prove sufficiency, first consider the (trivial) case $T^s = \{(0,0)\}$. Then, every party simply broadcasts its inputs and computes the function on the broadcasted values (c.f. Section 2.4). Otherwise, we use the protocol described in Figure 5 with $d = \hat{t_p^s}$, where $\hat{t_p^s}$ is the maximal t_p^s value in T^s. Correctness is always guaranteed, and Secrecy follows immediately from the choice of d.
Robustness: Since $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^r$ and $\forall (t_a^r, \cdot) \leq T^r, (\cdot, t_p^s) \leq T^s : t_a^r + t_p^s < n$, we have that $\forall (\cdot, t_p^s) \leq T^s : |\mathcal{D}^*| + t_p^s < n$. Then, it follows from the choice of d that $|\mathcal{D}^*| + d < n$.
Fairness: Given is that $(|\mathcal{D}^*|, |\mathcal{E}^*|) \leq T^f$. Since $T^f \leq T^s$, we have $|\mathcal{E}^*| \leq d$. Furthermore, since $\forall (t_a^f, t_p^f) \leq T^f : t_a^f + t_p^f < n$, we immediately have $|\mathcal{D}^*| + |\mathcal{E}^*| < n$.
The proof of necessity is given in the next section.

5.3 Proof of Necessity for Reactive MPC

In this section, we prove that the bound in Theorem 3 is necessary, i.e. if violated, (reactive) MPC is impossible. The bound in Theorem 3 is violated if $T^s \neq \{(0,0)\}$ and

$$\exists (t_a^r, \cdot) \leq T^r, (\cdot, t_p^s) \leq T^s : \quad t_a^r + t_p^s \geq n \quad \vee \quad \exists (t_a^f, t_p^f) \leq T^f : \quad t_a^f + t_p^f \geq n$$

Case: $\exists(t_a^r, \cdot) \leq T^r, (\cdot, t_p^s) \leq T^s : \quad t_a^r + t_p^s \geq n$. Assume that there is a secure protocol for this setting. Then, the adversary corrupts $(\mathcal{D}^*, \mathcal{E}^*) = (\mathcal{D}, \mathcal{D})$ with $\mathcal{D} \subseteq \mathcal{P}$ and $|\mathcal{D}| = n - t_p^s$ and has the parties in $\mathcal{D}$ stop sending messages. Since there are only t_p^s remaining parties, the state is lost and the computation cannot be continued. Hence, robustness is violated.[11]

Case: $\exists(t_a^f, t_p^f) \leq T^f : \quad t_a^f + t_p^f \geq n$. Same as in the non-reactive case (Section 4.3).

6 Conclusions

In this work, we have generalized and extended known results from the literature. In particular, we improved over the work in [IKLP06] that combines optimal results from the active and the passive world. Our protocols distinguish not only whether or not active cheating occurs, but provides a dynamic tradeoff between active and passive corruptions. Hence, we achieve "the best of both worlds – and everything in between" with a single protocol.

Furthermore, we introduced the notion of *gradual* verifiable secret sharing. This notion requires that, during reconstruction, secrecy is given up gradually, one subset at a time, while immediately establishing robustness against the corresponding complement set. As a consequence, intuitively speaking, the adversary might still abort the protocol, but does not automatically learn the secret. This technique turned out to be very useful in the setting of both non-reactive and reactive MPC to provide more flexible and therefore more practical protocols.

Moreover, the use of multi-thresholds allows to unify two incomparable models for combining active and passive corruption. In the first model, used for example by [IKLP06], the adversary can corrupt parties either passively or actively, but not both at the same time. Then, for each of the two corruption options, a maximally tolerable adversary is considered. In the second model, used for example by [FHM98], the adversary can corrupt some parties actively, and additionally some parties passively, at the same time. Yet, their model only allows to consider a single maximally tolerable adversary. By using multi-thresholds, we can provide a single protocol that subsumes results for both models simultaneously.

A Comparison with Related Work

For completeness, we summarize the considered models and achieved security levels of several protocols in the literature. In case of protocols with hybrid security, we indicate in parentheses over which properties the hybridization is achieved.

[11] Note that the proof in [IKLP06] considers only the special case where $t_a^r \leq t_p^s$.

Paper	Prot	Adv.	Security
[Cha89]	MPC	mixed	hybrid (comp/statistical)
[DDWY93]	SMT	mixed	perfect
[FHM98]	MPC	mixed	statistical
[HMZ08]	MPC	mixed	computational or statistical
[FHHW03]	BA	active	perfect
[FHW04]	MPC	active	hybrid (comp/stat)
[Kat07]	MPC	active	hybrid (output guarantee)
[LRM10]	MPC	active	hybrid (comp/stat and robustness/fairness)
[HLMR11]	MPC	mixed	perf., hybrid (privacy/correctness/robustness/fairness)
[HLMR12]	MPC	mixed	stat., hybrid (privacy/correctness/robustness/fairness)
this work	MPC	mixed	comp., hybrid (privacy/correctness/robustness/fairness)

MPC/SMT/BA = MPC/secure message transmission/Byzantine agreement, comp/stat/perf = computational/statistical/perfect.

References

[Bea91] Beaver, D.: Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority. Journal of Cryptology 4(2), 75–122 (1991)

[BGW88] Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: STOC 1988, pp. 1–10. ACM (1988)

[Blu83] Blum, M.: How to exchange (secret) keys (extended abstract). In: STOC 1983, pp. 440–447. ACM (1983)

[CCD88] Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols. In: STOC 1988, pp. 11–19. ACM (1988)

[Cha89] Chaum, D.: The spymasters double-agent problem: Multiparty computations secure unconditionally from minorities and cryptograhically from majorities. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 591–602. Springer, Heidelberg (1990)

[DDWY93] Dolev, D., Dwork, C., Waarts, O., Yung, M.: Perfectly secure message transmission. Journal of the ACM 40(1), 17–47 (1993)

[DS82] Dolev, D., Strong, H.R.: Polynomial algorithms for multiple processor agreement. In: STOC 1982, pp. 401–407. ACM (1982)

[FHHW03] Fitzi, M., Hirt, M., Holenstein, T., Wullschleger, J.: Two-threshold broadcast and detectable multi-party computation. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 51–67. Springer, Heidelberg (2003)

[FHM98] Fitzi, M., Hirt, M., Maurer, U.M.: Trading correctness for privacy in unconditional multi-party computation (extended abstract). In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 121–136. Springer, Heidelberg (1998)

[FHW04] Fitzi, M., Holenstein, T., Wullschleger, J.: Multi-party computation with hybrid security. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 419–438. Springer, Heidelberg (2004)

[GMW87] Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: STOC 1987, pp. 218–229. ACM (1987)

[Gol04] Goldreich, O.: Foundations of Cryptography. Basic Applications, vol. 2. Cambridge University Press (2004)

[HLMR11] Hirt, M., Lucas, C., Maurer, U., Raub, D.: Graceful degradation in multi-party computation (extended abstract). In: Fehr, S. (ed.) ICITS 2011. LNCS, vol. 6673, pp. 163–180. Springer, Heidelberg (2011)

[HLMR12] Hirt, M., Lucas, C., Maurer, U., Raub, D.: Passive corruption in statistical multi-party computation. In: Smith, A. (ed.) ICITS 2012. LNCS, vol. 7412, pp. 129–146. Springer, Heidelberg (2012)

[HMZ08] Hirt, M., Maurer, U., Zikas, V.: MPC vs. SFE: Unconditional and computational security. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 1–18. Springer, Heidelberg (2008)

[IKLP06] Ishai, Y., Kushilevitz, E., Lindell, Y., Petrank, E.: On combining privacy with guaranteed output delivery in secure multiparty computation. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 483–500. Springer, Heidelberg (2006)

[Kat07] Katz, J.: On achieving the "best of both worlds" in secure multiparty computation. In: STOC 2007, pp. 11–20. ACM (2007)

[LRM10] Lucas, C., Raub, D., Maurer, U.: Hybrid-secure MPC: Trading information-theoretic robustness for computational privacy. In: PODC 2010, pp. 219–228. ACM (2010)

[Ped91] Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992)

[RB89] Rabin, T., Ben-Or, M.: Verifiable secret sharing and multiparty protocols with honest majority. In: STOC 1989, pp. 73–85. ACM (1989)

[Sha79] Shamir, A.: How to share a secret. Communications of the ACM 22(11), 612–613 (1979)

[Yao82] Yao, A.C.: Protocols for secure computations (extended abstract). In: FOCS 1982, pp. 160–164. IEEE (1982)