General Adversaries in Unconditional Multi-party Computation[⋆]

Matthias Fitzi, Martin Hirt, and Ueli Maurer

Department of Computer Science
ETH Zurich
CH-8092 Zurich, Switzerland,
{fitzi,hirt,maurer}@inf.ethz.ch

Abstract. We consider a generalized adversary model for unconditionally secure multi-party computation. The adversary can actively corrupt (i.e. take full control over) a subset $D \subseteq P$ of the players, and, *additionally*, can passively corrupt (i.e. read the entire information of) another subset $E \subseteq P$ of the players. The adversary is characterized by a generalized adversary structure, i.e. a set of pairs (D, E), where he may select one arbitrary pair from the structure and corrupt the players accordingly. This generalizes the classical threshold results of Ben-Or, Goldwasser and Wigderson, Chaum, Crépeau, and Damgård, and Rabin and Ben-Or, and the non-threshold results of Hirt and Maurer.

The generalizations and improvements on the results of Hirt and Maurer are three-fold: First, we generalize their model by considering mixed (active and passive) non-threshold adversaries and characterize completely the adversary structures for which unconditionally secure multi-party computation is possible, for four different models: Perfect security with and without broadcast, and unconditional security (with negligible error probability) with and without broadcast. All bounds are tight. Second, some of their protocols have complexity super-polynomial in the size of the adversary structure; we reduce the complexity to polynomial. Third, we prove the existence of adversary structures for which no polynomial (in the number of players) protocols exist.

The following two implications illustrate the usefulness of these results: The most powerful adversary that is unconditionally tolerated by previous protocols among three players is the one that passively corrupts one arbitrary player; using our protocols one can unconditionally tolerate an adversary that either passively corrupts the first player, or *actively* corrupts the second or the third player.

Moreover, in a setting with arbitrarily many cheating players who want to compute an agreed function with the help of a trusted party, we can relax the trust requirement into this helping party: Without support from the cheating players the helping party obtains no information about the honest players' inputs and outputs.

Keyword: General adversaries, mixed model, multi-party computation, unconditional security.

⋆ Research supported by the Swiss National Science Foundation (SNF), project no. SPP 5003-045293.

1 Introduction

1.1 Secure Multi-Party Computation

Consider a set of n players who do not trust each other. Nevertheless they want to compute an agreed function of their inputs in a secure way. Security means achieving correctness of the result of the computation while keeping the players' inputs private, even if some of the players are corrupted by an adversary. This is the well-known secure multi-party computation problem, as first stated by Yao [Yao82].

As the first general solution to this problem, Goldreich, Micali, and Wigderson [GMW87] presented a protocol that allows n players to securely compute an arbitrary function even if an adversary actively corrupts any $t < n/2$ of the players and makes them misbehave maliciously. However, this protocol assumes that the adversary is computationally bounded. In a model with secure and authenticated channels between each pair of players (the secure-channels model), Ben-Or, Goldwasser, and Wigderson [BGW88], and Chaum, Crépeau, and Damgård [CCD88] proved that unconditional security is possible if at most $t < n/2$ of the players are passively corrupted, or alternatively, if at most $t < n/3$ of the players are actively corrupted. The bound $t < n/3$ for the active model was improved by Rabin and Ben-Or [RB89], Beaver [Bea91], and Cramer, Damgård, Dziembowski, Hirt, and Rabin [CDD$^+$99] to $t < n/2$ by assuming the existence of a broadcast channel.

Secure multi-party computation can alternatively, and more generally, be seen as the problem of performing a task among a set of players. The task is specified by involving a trusted party, and the goal of the protocol is to replace the need for the trusted party. In other words, the functionality of the trusted party is shared among the players. Secure function evaluation described above can be seen as a special case of this more general setting. Most protocols described in the literature in the context of secure function evaluation also apply in the general context. This also holds for the protocols described in this paper.

1.2 General Adversaries

Ito, Saito, and Nishizeki [ISN87] and Benaloh and Leichter [BL88] introduced the notion of general (non-threshold) access structures for secret sharing. For a set P of players, an *access structure* Γ is the set of all subsets of P that are qualified to reconstruct the secret. Hirt and Maurer [HM97] transferred and adjusted this notion to the field of general multi-party computation: for a set P of players, an *adversary structure* $\mathcal{Z}$ is a set of all subsets of P that are tolerated to jointly cheat without violating the security of the computation. A multi-party computation protocol is called $\mathcal{Z}$-secure if its security is not affected even if an adversary corrupts the players in one particular set in $\mathcal{Z}$.

1.3 Contributions

The main results of [HM97] state that in the passive model, every function can be computed unconditionally $\mathcal{Z}$-securely if and only if no *two* sets in $\mathcal{Z}$ cover the full set P of players. In the active model, every function can be computed $\mathcal{Z}$-securely if and only if no *three* sets in $\mathcal{Z}$ cover P. Assuming the existence of broadcast channels and allowing some negligible error probability, every function can be computed $\mathcal{Z}$-securely if and only if no *two* sets in $\mathcal{Z}$ cover P.

We unify these models and introduce a new model in which the adversary may actively corrupt some players, and, *at the same time*, passively corrupt some additional players. The adversary is characterized by a generalized adversary structure, a set of *classes* (D, E) of subsets of the player set P (i.e. $D, E \subseteq P$), where the players of one specific class (D, E) in the adversary structure may be corrupted — actively for the players in D (disruption) and passively for the players in E (eavesdropping).

For example, the adversary structure $\mathcal{Z} = \big\{(\{p_1\}, \{p_2, p_3\}), (\{p_2\}, \{p_4\})\big\}$ describes an adversary that may *either* simultaneously corrupt player p_1 actively and the players p_2 and p_3 passively, *or* simultaneously corrupt player p_2 actively and player p_4 passively. Note that it is not known in advance which class of the structure will be corrupted by the adversary (and this is typically even not known at the end of the protocol).

For this unified model, the necessary and sufficient conditions for secure multi-party computation to be achievable for all functions are derived. In order to characterize these conditions, we introduce three predicates: Let P be a set of players and let $\mathcal{Z}$ be an adversary structure for P. Then $Q^{(2,2)}(P, \mathcal{Z})$ is the predicate that is satisfied if and only if the players of no two classes in $\mathcal{Z}$ cover the full set P of players, $Q^{(3,2)}(P, \mathcal{Z})$ is the predicate that is satisfied if and only if the players of no two classes in $\mathcal{Z}$ together with the players in the active set of any other class in $\mathcal{Z}$ cover P, and finally, $Q^{(3,0)}(P, \mathcal{Z})$ is the predicate that is satisfied if and only if the players in the active sets of any three classes in $\mathcal{Z}$ do not cover P. Formally,

$$Q^{(2,2)}(P, \mathcal{Z}) \iff \forall (D_1, E_1), (D_2, E_2) \in \mathcal{Z} : D_1 \cup E_1 \cup D_2 \cup E_2 \neq P \ ,$$

$$Q^{(3,2)}(P, \mathcal{Z}) \iff \forall (D_1, E_1), (D_2, E_2), (D_3, E_3) \in \mathcal{Z} : D_1 \cup E_1 \cup D_2 \cup E_2 \cup D_3 \neq P \ ,$$

$$Q^{(3,0)}(P, \mathcal{Z}) \iff \forall (D_1, E_2), (D_2, E_2), (D_3, E_3) \in \mathcal{Z} : D_1 \cup D_2 \cup D_3 \neq P \ .$$

We characterize the necessary and sufficient conditions on the existence of unconditionally secure multi-party protocols according to three different cases:

– With or without broadcast channels, perfectly secure (without any probability of error) multi-party computation is achievable if and only if $Q^{(3,2)}(P, \mathcal{Z})$ is satisfied.
– Given a broadcast channel, unconditionally secure (with negligible probability of error) multi-party computation is achievable if and only if $Q^{(2,2)}(P, \mathcal{Z})$ is satisfied.

– Without a broadcast channel, unconditionally secure multi-party computation is achievable if and only if both predicates $Q^{(2,2)}(P, \mathcal{Z})$ and $Q^{(3,0)}(P, \mathcal{Z})$ are satisfied.

Moreover, for all models we propose constructions that yield protocols with computation and communication complexity polynomial in the size of the adversary structure and linear in the size of the circuit computing the function, as opposed to the protocols of [HM97] that have super-polynomial complexity in those cases with error probability. Furthermore we show that this construction is optimal in the sense that there are adversary structures which *require* protocols with complexity at least polynomial in the size of the adversary structure (and hence potentially super-polynomial in the number of players).

1.4 Related Work

Active and passive corruptions within the same model was first considered by Galil, Haber, and Yung [GHY87] for threshold multi-party computation. Chaum [Cha89] proposed protocols that provide security with respect to an adversary that either passively or actively corrupts players up to given thresholds. Dolev, Dwork, Waarts, and Yung [DDWY93] proposed protocols and proved tight bounds for message transmission unconditionally secure in simultaneous presence of active and passive corruptions.

Fitzi, Hirt, and Maurer [FHM98] proposed multi-party protocols secure against mixed *threshold* adversaries. Based on the constructions of classical multi-party protocols [BGW88,RB89], they constructed new protocols for an adversary that simultaneously actively, passively, and fail-corrupts players up to given thresholds. However, as pointed out by Damgård [Dam99], their protocols for the perfect model (without error probability) do not achieve security for all thresholds within the claimed bounds.[1] In contrast to their work that modified existing protocols in order to achieve the required properties, in this paper we use the technique of *player simulation* [HM97] with classical protocols [BGW88,RB89] as a basis.

Cramer, Damgård, and Maurer [CDM99] proved that for every adversary structure for which multi-party computation is feasible and for which there is an efficient linear secret-sharing scheme, efficient multi-party protocols exist. Smith and Stiglic [SS98] consider also uniquely active adversaries and propose protocols for the active model with broadcast. The efficiency of their protocols is polynomial in the size of a span program that computes the adversary structure, however in Section 4 we prove that for some adversary structures, *every* protocol requires complexity exponential in the number of players. This proof also applies to models with only passive or only active corruptions.

[1] Indeed, the tightness proofs for the perfect models in this paper contradict the results of [FHM98]. See [Dam99] for more details.

1.5 Outline

In Sect. 2 we formally define the models. The main results of the paper, the characterization of the exact conditions for secure multi-party protocols as well as the protocol constructions, are given in Sect. 3. In Sect. 4 we prove the existence of adversary structures for which no protocols with complexity polynomial in the number of players exist. Finally, some conclusions and open problems are mentioned in Sect. 5.

2 Definitions and Model

This section gives a formal definition of the model used in this paper.

2.1 Protocols

A *processor* can perform operations in a fixed finite field $(\mathcal{F}, +, *)$, can select elements from this field at random, and can communicate with other processors over perfectly authenticated and confidential synchronous channels (secure channels model).[2]

A *protocol* π among a set P of processors is a sequence of statements. There are input and output statements, transmit statements, and computation statements. The latter include addition, multiplication, and random selection of field elements.

A *multi-party computation specification* (or simply called specification) formally describes the cooperation to be performed. Intuitively, a specification specifies the cooperation in an ideal environment involving a trusted party. Formally, a specification is a pair (π_0, τ) consisting of a protocol π_0 among a set P_0 of processors, and the name of a trusted processor $\tau \in P_0$.

A general approach to multi-party computation is to construct protocols for arbitrary specifications, or, more generally, to find a function (called *multi-party protocol generator*) that takes a specification as an input and outputs a protocol that securely computes the specification.

2.2 Adversaries and Adversary Structures

An adversary A is a program that actively corrupts a certain subset of the processors and, at the same time, passively corrupts another subset of the processors. To passively corrupt a processor means to be able to permanently read *all* variables of that processor. To actively corrupt a processor means to be able to take full control over the processor, in particular to read and write all its

[2] In contrast to the players mentioned in the introduction, a processor is considered to only perform the computation, where inputs and outputs are given from/to some other entity (e.g. a person). This distinction avoids misunderstandings when processors are simulated by multi-party protocols.

variables. The complexity of an adversary is not assumed to be polynomial and may be unlimited.

An adversary is characterized by an *adversary class* $C = (D, E)$, a pair of disjoint subsets of the processor set, i.e. $D, E \subseteq P$ and $D \cap E = \emptyset$. An adversary of class (D, E) may actively corrupt the processors in D (disruption) and may passively corrupt the processors in E (eavesdropping). The set D is called the active set and the set E is called the passive set of the class. A processor is *contained* in an adversary class if it is a member in either set, i.e. $p \in (D, E) \Leftrightarrow p \in (D \cup E)$. An adversary class C' is contained in an adversary class C if the active set of C' is a subset of the active set of C, and if every processor contained in C' is also contained in C, i.e. $(D', E') \subseteq (D, E) \Leftrightarrow (D' \subseteq D) \wedge (E' \subseteq (D \cup E))$.[3] An adversary class C' is *strictly contained* in an adversary class C if it is contained but not equal, i.e. $C' \subset C \Leftrightarrow (C' \subseteq C \wedge C' \neq C)$. An *adversary structure* $\mathcal{Z}$ for the set P of processors is a monotone set of adversary classes, i.e. for every class $C \in \mathcal{Z}$, all classes contained in C are also in $\mathcal{Z}$. For a structure $\mathcal{Z}$, $\overline{\mathcal{Z}}$ denotes the *basis* of the structure, i.e. the set of the maximal classes in $\mathcal{Z}$: $\overline{\mathcal{Z}} = \{C \in \mathcal{Z} : \nexists C' \in \mathcal{Z} : C \subset C'\}$.

To *restrict* a class $C = (D, E)$ to the set P' of processors, denoted $(D, E)\big|_{P'}$, means to intersect both sets of the class with P', i.e. $(D, E)\big|_{P'} = (D \cap P', E \cap P')$. To restrict a structure $\mathcal{Z}$ to the set P' of processors means to restrict all classes in the structure.

2.3 Security

For an adversary A, a protocol *A-securely computes* a specification if, whatever A does in the protocol, the same effect could be achieved by A (with a modified strategy, but with similar costs) in the specification [Can98,Bea91,MR98]. For an adversary structure $\mathcal{Z}$ and a specification (π_0, τ), a protocol π *$\mathcal{Z}$-securely computes* the specification (π_0, τ) if for every adversary A of class $C \in \mathcal{Z}$, the protocol π A-securely computes the specification (π_0, τ). Whenever the specification is clear from the context, we also say that a protocol *tolerates* an adversary A (a structure $\mathcal{Z}$) instead of saying that the protocol A-securely ($\mathcal{Z}$-securely) computes the specification.

3 Complete Characterization of Tolerable Adversaries

The basic technique for constructing a protocol that tolerates a given adversary structure is to begin with a threshold protocol (e.g. one of the protocols of [BGW88,CCD88,RB89]) among a small number of processors and to simulate some of these processors by subprotocols among appropriate sets of other processors [HM97]. The idea behind this is that everything a processor has to perform during the protocol execution (such as communication with other processors and local computations) can be simulated by a multi-party computation

[3] This definition implies that every adversary of a given class C' can also be considered as an adversary of every class C with $C' \subseteq C$.

protocol among a set of processors. If the adversary is tolerated by this simulation protocol then the simulated processor can be considered to be uncorrupted. This procedure of processor simulation can be applied recursively, i.e., the processors that participate in the simulation of a processor can again be simulated by an appropriate set of other processors, and so on.

The proofs given in this section are only sketched. Formal proofs based on simulator techniques can be given according to [Can98,Bea91,MR98]. Also, the proofs in this section are given with respect to a static adversary (i.e. an adversary that at the beginning of the protocol selects the processors to be corrupted), but they can be easily modified to apply to a model with an adaptive adversary (i.e. an adversary that consecutively corrupts processors during the computation, depending on the information gained so far, where the processors corrupted at any time must form an admissible class in the adversary structure).

3.1 Perfectly Secure Multi-Party Computation

The main result of this section, the tight bounds as well as the protocol construction, are stated in Theorem 1. This general result is based on a solution for all adversary structures $\mathcal{Z}$ with $|\overline{\mathcal{Z}}| \leq 3$, which is given in the following lemma.

Lemma 1. *A set P of processors can compute every function/specification perfectly $\mathcal{Z}$-securely if $Q^{(3,2)}(P, \mathcal{Z})$ and $|\overline{\mathcal{Z}}| \leq 3$. The computation and communication complexities are linear in the size of the specification.*

Proof. Consider an arbitrary adversary structure $\mathcal{Z}$ with $|\overline{\mathcal{Z}}| \leq 3$ that satisfies $Q^{(3,2)}(P, \mathcal{Z})$, and a specification (π_0, τ). We show that for every such structure $\mathcal{Z}$ there exists a subset of the processors that can compute the specification in a secure way.[4] If $|\overline{\mathcal{Z}}| < 3$, then the condition $Q^{(3,2)}(P, \mathcal{Z})$ immediately implies that there is a processor $p \in P$ that is not contained in any class of $\overline{\mathcal{Z}}$ (i.e. $\mathcal{Z}\big|_{\{p\}} = \{(\emptyset, \emptyset)\}$). Hence this processor cannot be corrupted by any admissible adversary, and one can simply replace the occurrence of the trusted party τ in the protocol π_0 of the specification by the name of this processor. Thus assume that $|\overline{\mathcal{Z}}| = 3$ and $\overline{\mathcal{Z}} = \{(D_1, E_1), (D_2, E_2), (D_3, E_3)\}$. Condition $Q^{(3,2)}(P, \mathcal{Z})$ implies that there exists a processor $p_3 \in P$ with $p_3 \notin D_1 \cup E_1 \cup D_2 \cup E_2 \cup D_3$ (but potentially $p_3 \in E_3$). Hence this processor remains uncorrupted by an adversary of the first or the second class, and is (at most) passively corrupted by an adversary of the third class. By symmetry reasons, there exist processors p_1 and p_2 which can be corrupted at most passively and only by an adversary of the first or the second class, respectively. This means that every admissible adversary may corrupt none of the processors p_1, p_2, or p_3 actively and only at most one of them passively. Hence, these three processors can simulate the trusted party of the specification by using the protocol of [BGW88] (passive model) for three processors. The other processors (if any) are not involved in the simulation of the trusted party. □

[4] Although only a subset of the processors is involved in the multi-party computation, all the processors that have input must provide (i.e. secret-share) this input among the involved processors.

Theorem 1. *A set P of processors can compute every function/specification perfectly $\mathcal{Z}$-securely if $Q^{(3,2)}(P,\mathcal{Z})$ is satisfied. This bound is tight: if $Q^{(3,2)}(P,\mathcal{Z})$ is not satisfied, then there exist functions that cannot be computed perfectly $\mathcal{Z}$-securely, even if a broadcast channel is available.*[5] *The communication and computation complexities are polynomial in the size $|\overline{\mathcal{Z}}|$ of the basis of the adversary structure and linear in the length of the specification.*

Proof. Consider a set P of processors and a structure $\mathcal{Z}$ for this set P such that $Q^{(3,2)}(P,\mathcal{Z})$ is satisfied, and an arbitrary specification (π_0,τ). We recursively construct a $\mathcal{Z}$-secure protocol π:

The case $|\overline{\mathcal{Z}}| \leq 3$ was treated in Lemma 1 (induction basis). Thus assume that $|\overline{\mathcal{Z}}| \geq 4$, and that for all adversary structures with basis size strictly less than k there exists a secure protocol (induction hypothesis). We select some four-partition of $\overline{\mathcal{Z}}$ where the size of each set of the partition is at least $\lfloor|\overline{\mathcal{Z}}|/4\rfloor$. Let $\mathcal{Z}_1$, $\mathcal{Z}_2$, $\mathcal{Z}_3$, and $\mathcal{Z}_4$ be the four unions of three distinct sets of the partition, each of them completed such that it is monotone. Since $|\overline{\mathcal{Z}}| \geq 4$, the size $|\overline{\mathcal{Z}_i}|$ of the basis of each such structure is strictly smaller than the size $|\overline{\mathcal{Z}}|$ of the current structure basis, i.e. $|\overline{\mathcal{Z}_i}| < |\overline{\mathcal{Z}}|$ $(1 \leq i \leq 4)$, and one can recursively construct protocols π_1, π_2, π_3, and π_4, each among the set P of processors, tolerating $\mathcal{Z}_1$, $\mathcal{Z}_2$, $\mathcal{Z}_3$, and $\mathcal{Z}_4$, respectively (hypothesis). The protocol π that tolerates $\mathcal{Z}$ can be constructed as follows:

First, one constructs a protocol among four "virtual" processors that computes the specification (π_0,τ), tolerating an adversary that actively corrupts a single processor [BGW88] (active model). Then one simulates the four virtual processors by the recursively constructed protocols $\pi_1, \ldots, \pi_4$, respectively. Since every adversary class is tolerated by at least three of the protocols π_1, π_2, π_3, and π_4 (thus only one of the virtual processors in the main protocol is misbehaving), the resulting protocol tolerates all adversary classes in the adversary structure and, as claimed, the constructed protocol π is $\mathcal{Z}$-secure.

In order to analyze the efficiency of the protocols, we need the help of the following observation: The protocols of [BGW88] for the passive model with three processors and those for the active model with four processors have a constant "blow-up factor" b_p and b_a, respectively, i.e. for any specification of length l, the length of the protocol computing this specification is bounded by $b_p \cdot l$ in the passive model and by $b_a \cdot l$ in the active model.

In the construction given above, on each recursion level all involved processors are simulated by using protocols of [BGW88] (active case), except for the lowest level, where [BGW88] (passive case) is used. The simulations on each level can be performed independently, and every statement in the current protocol is affected by at most two such simulations (as at most two processors occur in one statement). Hence, the total blow-up of all simulations on a given level is bounded by b_a^2 (b_p^2 on the lowest level), and as the recursion depth of the construction is logarithmic in the number $|\overline{\mathcal{Z}}|$ of maximal sets in the adversary structure, the total blow-up is polynomial in $|\overline{\mathcal{Z}}|$.

[5] Indeed, almost every non-trivial function cannot be computed perfectly $\mathcal{Z}$-securely.

In order to prove the tightness of the theorem, assume an adversary structure $\mathcal{Z}$ for which every function can be computed perfectly $\mathcal{Z}$-securely and suppose $Q^{(3,2)}(P, \mathcal{Z})$ is not satisfied. Then there exist three classes $(D_1, E_1), (D_2, E_2), (D_3, E_3) \in \mathcal{Z}$ with $D_1 \cup E_1 \cup D_2 \cup E_2 \cup D_3 = P$, and (due to the monotonicity of $\mathcal{Z}$) with the sets D_1, E_1, D_2, E_2 and D_3 being pairwise disjoint.

One can construct a protocol for three processors $\hat{p}_1$, $\hat{p}_2$, and $\hat{p}_3$, where $\hat{p}_1$ plays for all the processors in $D_1 \cup E_1$, $\hat{p}_2$ plays for those in $D_2 \cup E_2$, and $\hat{p}_3$ plays for those in D_3. This new protocol is secure with respect to an adversary that passively corrupts either $\hat{p}_1$ or $\hat{p}_2$, or actively corrupts $\hat{p}_3$.

Assume that the specification requires to compute the logical `AND` of two bits x_1 and x_2 held by $\hat{p}_1$ and $\hat{p}_2$, respectively, and assume for the sake of contradiction that a protocol for this specification is given. Let T denote the transcript of the broadcast channel of a run of that protocol (if no broadcast channel is available, let $T = \emptyset$), and let T_{ij} $(1 \leq i < j \leq 3)$ denote the transcript of the channels between $\hat{p}_i$ and $\hat{p}_j$. Due to the requirement of perfect privacy, $\hat{p}_1$ will not send any information about his bit x_1 over T_{12} or over T before he knows x_2 (if P_1 knows that $x_2 = 1$ he can reveal x_1). Similarly, $\hat{p}_2$ will not send any information about x_2 over T_{12} or over T before he knows x_1. Hence the only escape from this deadlock would be to use $\hat{p}_3$. However, as T_{12} and T jointly give no information about x_2, a random misbehavior of an actively corrupted $\hat{p}_3$ (ignore all received messages and send random bits whenever a message must be sent) would with some (possibly negligible) probability make $\hat{p}_1$ receive the wrong output, contradicting the perfect security of the protocol. □

3.2 Unconditionally Secure Multi-Party Computation

We prove the necessity of $Q^{(2,2)}$ for unconditionally secure multi-party computation in Lemma 2, and prove its sufficiency for the case that broadcast channels are available in Theorem 2. We then consider a model without broadcast and suggest a simple but surprising protocol among three processors for this model (Theorem 3). Finally, in Theorem 4, the tight bounds on the existence of unconditionally secure multi-party protocols in a model without broadcast are given. Note that all proposed protocols are efficient (polynomial in the number of maximal sets in the adversary structure), as opposed to the protocols for the unconditional model in [HM97].

Lemma 2. *For every adversary structure $\mathcal{Z}$ for a processor set P not satisfying $Q^{(2,2)}(P, \mathcal{Z})$, there exist functions/specifications that cannot be computed unconditionally $\mathcal{Z}$-securely. Even a broadcast channel does not help.*

Proof. For the sake of contradiction, assume that for an adversary structure $\mathcal{Z}$ for which $Q^{(2,2)}(P, \mathcal{Z})$ is not satisfied, there exists an unconditional $\mathcal{Z}$-secure protocol for every function. There exist two classes $(D_1, E_1), (D_2, E_2) \in \mathcal{Z}$ with $D_1 \cup E_1 \cup D_2 \cup E_2 = P$. Without loss of generality, assume that the four sets D_1, E_1, D_2, and E_2 are pairwise disjoint. Then we can transform such a $\mathcal{Z}$-secure protocol into a protocol among two processors $\hat{p}_1$ and $\hat{p}_2$, where each processor plays for the processors in $D_1 \cup E_1$, and $D_2 \cup E_2$, respectively. The broadcast channel is not needed anymore (there are only two processors). This protocol is secure against passive corruption of one of the two processors, contradicting Theorem 2 of [BGW88]. □

Theorem 2. *If a broadcast channel is available, a set P of processors can compute every function/specification unconditionally $\mathcal{Z}$-securely if $Q^{(2,2)}(P, \mathcal{Z})$ is satisfied. This bound is tight: if $Q^{(2,2)}(P, \mathcal{Z})$ is not satisfied, then there exist functions that cannot be computed unconditionally $\mathcal{Z}$-securely. The communication and computation complexities of the protocol are polynomial in the size $|\overline{\mathcal{Z}}|$ of the basis of the adversary structure and linear in the length of the specification.*

Proof. Consider a set P of processors and a structure $\mathcal{Z}$ for this set P such that $Q^{(2,2)}(P, \mathcal{Z})$ is satisfied, and an arbitrary specification (π_0, τ). We have to construct a $\mathcal{Z}$-secure protocol π for the set P of processors.

The case $|\overline{\mathcal{Z}}| \leq 3$ is simple. Since we have $Q^{(2,2)}(P, \mathcal{Z})$, we have three processors p_1, p_2, and p_3, where p_i occurs in the i-th class of $\mathcal{Z}$, but does not occur in the other classes. The protocol of [RB89] for three processors requires a broadcast channel and provides unconditional security (with some negligible error probability) with respect to an adversary that actively corrupts a single processor (trivially, this processor may also be corrupted only passively). This protocol among the three processors p_1, p_2, and p_3 is $\mathcal{Z}$-secure.

The case of a basis with at least four classes is treated along the lines of the construction in the proof of Theorem 1: First we select some four-partition of $\overline{\mathcal{Z}}$ and, by recursion, a protocol is constructed for each of the four unions of three subsets of the partition. Then, these four protocols are composed to a four-party protocol of [BGW88, active model].

The efficiency of this protocol can be analysed along the lines of the analysis given in the proof of Theorem 1. However, as the protocols of [RB89] that are used in the lowest level of the substitution tree provide some negligible error probability, special care is required in the analysis (cf. [HM97]). It follows immediately from the analysis in the proof of Theorem 1 that the protocol which results after applying all substitutions except for those on the lowest level, has polynomial complexity. But every statement of this protocol is expanded at most twice by all the remaining substitutions (once per involved processor), and each blow-up is polynomial, and hence the final protocol is also polynomial in the number $|\overline{\mathcal{Z}}|$ of maximal sets in the adversary structure. This is in contrast to the protocols of [HM97] (for the unconditional model with error probability), where protocols of [RB89] are used in each level of the simulation tree and hence their protocols have superpolynomial complexity.

The tightness of the theorem is given in Lemma 2. □

Proposition 1. *Let $\mathcal{Z}$ be an adversary structure for the set P of processors, where one processor $p \in P$ does not occur in the active set of any class $C \in \mathcal{Z}$ (i.e. $\forall (D, E) \in \mathcal{Z} : p \notin D$). If there exists a $\mathcal{Z}$-secure protocol π in a model with* *broadcast, then one can construct a $\mathcal{Z}$-secure protocol π' for a model without broadcast. The complexity of π' is the same as the complexity of π.*

Proof. Since there exists a processor $p \in P$ that cannot be actively corrupted by any admissible adversary, it is guaranteed that it follows the protocol. Hence, p can be used to simulate a broadcast channel. Instead of broadcasting a message, the message is sent to p which then sends this message to all processors in P. $\square$

Theorem 3. *A set $P = \{p_1, p_2, p_3\}$ of three processors can compute every function/specification unconditionally securely with respect to an adversary that either passively corrupts p_1 or actively corrupts either p_2 or p_3, i.e. $\mathcal{Z}$-securely for $\overline{\mathcal{Z}} = \big\{(\emptyset, \{p_1\}),\ (\{p_2\}, \emptyset),\ (\{p_3\}, \emptyset)\big\}$.*

Proof. In order to compute an arbitrary specification, the protocol of [RB89] is applied. This protocol for three processors provides unconditional security (with negligible error probability) with respect to an adversary that may actively corrupt one arbitrary processor, but it assumes the existence of a broadcast channel. However, the processor p_1 does not occur in the active set of any class in $\mathcal{Z}$, so, by Proposition 1, we can transform the protocol with a broadcast channel to a protocol that does not assume a broadcast channel. $\square$

Theorem 4. *A set P of processors can compute every function/specification unconditionally $\mathcal{Z}$-securely if $Q^{(2,2)}(P, \mathcal{Z})$ and $Q^{(3,0)}(P, \mathcal{Z})$ are satisfied. This bound is tight: if $Q^{(2,2)}(P, \mathcal{Z})$ or $Q^{(3,0)}(P, \mathcal{Z})$ is not satisfied, then there exist functions that cannot be computed unconditionally $\mathcal{Z}$-securely. The communication and computation complexities of the protocol are polynomial in the size $|\overline{\mathcal{Z}}|$ of the basis of the adversary structure and linear in the length of the specification.*

Proof. Consider a set P of processors and a structure $\mathcal{Z}$ for this set P such that $Q^{(2,2)}(P, \mathcal{Z})$ and $Q^{(3,0)}(P, \mathcal{Z})$ are satisfied. The condition $Q^{(3,0)}$ implies the existence of an efficient secure protocol for broadcast [FM98], and hence the construction of the proof of Theorem 2 yields a $\mathcal{Z}$-secure protocol.

The necessity of $Q^{(2,2)}(P, \mathcal{Z})$ was proven in Lemma 2. Thus assume that $Q^{(2,2)}(P, \mathcal{Z})$ is satisfied but not $Q^{(3,0)}(P, \mathcal{Z})$, i.e. there exist three classes $(D_1, E_1), (D_2, E_2), (D_3, E_3) \in \mathcal{Z}$ with $D_1 \cup D_2 \cup D_3 = P$ (and D_1, D_2, and D_3 pairwise disjoint). For the sake of contradiction, assume that for every function a $\mathcal{Z}$-secure multi-party protocol exists, hence in particular for the broadcast function. One can hence construct a broadcast protocol for the three processors $\hat{p}_1$, $\hat{p}_2$, and $\hat{p}_3$ (where each processor $\hat{p}_1$, $\hat{p}_2$, and $\hat{p}_3$ "plays" for the processors in one of the sets D_1, D_2, and D_3, respectively), where the adversary is allowed to actively corrupt one of them, contradicting the result that broadcast for three processors is not possible if the adversary may actively corrupt one of the processors, even if a negligible error probability is tolerated [LSP82,KY]. $\square$

Corollary 1. *Using the help of a trusted party τ, a set P of n processors can compute every function/specification unconditionally securely with respect to an adversary that may actively corrupt any subset $S \subseteq P$ of size $|S| \leq t$ (for a given*

$t > n/2$). The trusted party τ obtains no information about the private inputs and outputs as long as less than $n - t$ processors are actively corrupted.

Proof. We need to show that there are $\mathcal{Z}$-secure protocols for the set $P \cup \{\tau\}$ of processors, where $\mathcal{Z} = \{(D, E)\colon |D \cup E| \leq t\} \cup \{(D, E \cup \{\tau\})\colon |D \cup E| < n - t\}$. According to Theorem 4 it suffices to show that $Q^{(2,2)}(P \cup \{\tau\}, \mathcal{Z})$ and $Q^{(3,0)}(P \cup \{\tau\}, \mathcal{Z})$ are satisfied. Obviously, $Q^{(3,0)}(P \cup \{\tau\}, \mathcal{Z})$ holds since τ may not be actively corrupted.

In order to prove that $Q^{(2,2)}(P \cup \{\tau\}, \mathcal{Z})$ is satisfied, consider two arbitrary classes $(D_1, E_1), (D_2, E_2) \in \mathcal{Z}$. At least one of the classes must contain τ (else the classes cannot cover $P \cup \{\tau\}$), and this class has cardinality at most $n - t$. The other class has cardinality at most $n - t$ (if it contains τ) or t (if it does not contain τ), and the condition $t > n/2$ implies that in either case the sum cardinality of both classes is at most n. There are $n + 1$ processors in $P \cup \{\tau\}$, hence at least one processor does not occur in either class and $Q^{(2,2)}(P \cup \{\tau\}, \mathcal{Z})$ is satisfied. □

4 Adversary Structures without Efficient Protocols

The goal of this section is, informally, to prove that there exists a family of adversary structures for which the length of every resilient protocol grows exponentially in the number of processors.

For a specification (π_0, τ), a set P of processors, and an adversary structure $\mathcal{Z}$, let $\varphi\big((\pi_0, \tau), P, \mathcal{Z}\big)$ denote the length of the shortest protocol π for P that $\mathcal{Z}$-securely computes (π_0, τ). Furthermore, let (π_*, τ) denote the specification for the processors p_1 and p_2 that reads one input of both processors, computes the product and hands it to p_1. Finally, let P_n denote the set $\{p_1, \ldots, p_n\}$ of processors.

The following theorem shows that there exists a family $\mathcal{Z}_2, \mathcal{Z}_3, \ldots$ of adversary structures for the sets $P_2, P_3, \ldots$ of processors, respectively, such that $\varphi\big((\pi_*, \tau), P_n, \mathcal{Z}_n\big)$ grows exponentially in n.

Theorem 5. *For all considered models there is a family $\mathcal{Z}_2, \mathcal{Z}_3, \ldots$ of admissible adversary structures for the sets $P_2, P_3, \ldots$ of processors such that the length $\varphi\big((\pi_*, \tau), P_n, \mathcal{Z}_n\big)$ of the shortest $\mathcal{Z}_n$-secure protocol for (π_*, τ) grows exponentially in n.*

In order to prove the theorem we need an additional definition: An admissible adversary structure $\mathcal{Z}$ for the set P of processors is *maximal* if $Q^{(3,2)}(P, \mathcal{Z})$ is satisfied, but any adversary structure $\mathcal{Z}'$ with $\mathcal{Z} \subsetneq \mathcal{Z}'$ violates $Q^{(3,2)}(P, \mathcal{Z}')$.

Proof. The proof proceeds in three steps: First we prove that the number of maximal admissible adversary structures grows doubly-exponentially in the number n of processors. In the second step, we show that for the given specification

(π_*, τ), for every maximal admissible adversary structure a different protocol is required. Finally, we conclude that for some adversary structures the length of every secure protocol is exponential in the number of processors.

1. We exclusively consider adversary structures $\mathcal{Z}$ that only contain classes with an empty active set, i.e. $\forall(D,E) \in \mathcal{Z} : D = \emptyset$. Hence, the necessary and sufficient conditions for the existence of multi-party protocols is that the passive sets of no two classes in $\mathcal{Z}$ cover the full set P of processors. As a shorthand we write $E \in \mathcal{Z}$ instead of $(D,E) \in \mathcal{Z}$. Without loss of generality, assume that $n = |P|$ is odd, and let $m = (n+1)/2$. Fix a processor $p \in P$, and consider the set B that contains all subsets of $P \setminus \{p\}$ with exactly m processors, i.e. $B = \{E \subseteq (P\setminus\{p\}) \colon |E| = m\}$. For each subset $B' \subseteq B$, we define $\mathcal{Z}_{B'}$ to be the adversary structure that contains all sets in B', plus all sets $E \subseteq P$ with $|E| < m$ and $(P \setminus E) \notin B$. One can easily verify that $\mathcal{Z}_{B'}$ is admissible and maximal, and that for two different subsets $B', B'' \subseteq B$, the structures $\mathcal{Z}_{B'}$ and $\mathcal{Z}_{B''}$ are different. The size of B is $|B| = \binom{n-1}{m}$, hence there are $2^{\binom{n-1}{m}}$ different subsets B' of B, and thus doubly-exponentially many different maximal admissible adversary structures.
2. Let $\mathcal{Z}$ be a maximal admissible adversary structure, and let π be a protocol that $\mathcal{Z}$-securely computes (π_*, τ). For the sake of contradiction, assume that for some other maximal admissible adversary structure $\mathcal{Z}'$ (where $\mathcal{Z}' \neq \mathcal{Z}$), the same protocol π $\mathcal{Z}'$-securely computes (π_*, τ). Then π would $(\mathcal{Z} \cup \mathcal{Z}')$-securely compute (π_*, τ). However, since both $\mathcal{Z}$ and $\mathcal{Z}'$ are maximal admissible, $(\mathcal{Z} \cup \mathcal{Z}')$ is not admissible, and hence no such protocol exists. Hence, for each maximal admissible adversary structure $\mathcal{Z}$ a different protocol π is required for securely computing (π_*, τ).
3. There are doubly-exponentially many maximal admissible adversary structures, and for each of them, a different protocol is required, hence there are doubly-exponentially many different protocols. This concludes that some of these protocols have exponential length. $\square$

5 Conclusions and Open Problems

We have given a complete characterization of adversaries tolerable in unconditional multi-party computation in a generalized model where the adversary may actively corrupt some players and simultaneously passively corrupt some additional players. The characterization of the adversary is given by a set of pairs of subsets of the player set (rather than thresholds as in [Cha89,DDWY93,FHM98] or an adversary structure for either passive or active corruption [HM97,CDM99,SS98]). Moreover we have proposed constructions that, for any admissible adversary, yield secure protocols with communication and computation complexities polynomial in the size of the adversary structure. This improves on those protocols in [HM97] that have complexities super-polynomial in the size of the adversary structure.

For many scenarios, the protocols proposed in this paper tolerate strictly more powerful adversaries than are tolerated by any previous protocol. As a surprising example, the protocol for three players that unconditionally tolerates an adversary that passively corrupts one single player could be improved by tolerating that the adversary may corrupt one of two specific players even actively.

Finally, this paper has given a proof that there is a family of adversary structures which no protocol with complexities polynomial in the number of players exists for.

Besides active and passive player corruption, fail-corruption can be considered as a third fundamental type of player corruption, as treated in [GHY87,DDWY93,FHM98]. It is an open problem to characterize the tight conditions for unconditionally secure multi-party computation to be achievable with respect to a general adversary that may simultaneously perform active, passive and fail-corruptions. In the generalized adversary model this problem seems to be more involved than in the threshold model.

Acknowledgments

The authors would like to thank Ronald Cramer, Ivan Damgård, Yuval Ishai, and Stefan Wolf for many interesting discussions. Furthermore we would like to thank the anonymous referees for their useful comments on the paper.

References

Bea91. D. Beaver. Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority. *Journal of Cryptology*, pp. 75–122, 1991. 233, 237, 238

BGW88. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. 20th ACM Symposium on the Theory of Computing (STOC)*, pp. 1–10, 1988. 233, 235, 237, 238, 239, 241

BL88. J. C. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In *Advances in Cryptology — CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*. Springer-Verlag, 1988. 233

Can98. R. Canetti. Security and composition of multi-party cryptographic protocols. Manuscript, June 1998. Former (more general) version: Modular composition of multi-party cryptographic protocols, Nov. 1997. 237, 238

CCD88. D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proc. 20th ACM Symposium on the Theory of Computing (STOC)*, pp. 11–19, 1988. 233, 237

CDD$^+$99. R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations with dishonest minority. In *Advances in Cryptology — EUROCRYPT '99*, Lecture Notes in Computer Science, 1999. 233

CDM99. R. Cramer, I. Damgård, and U. Maurer. General secure multi-party computation from any linear secret sharing scheme. Manuscript, 1999. 235, 244

Cha89. D. Chaum. The spymasters double-agent problem. In *Advances in Cryptology — CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pp. 591–602. Springer-Verlag, 1989. 235, 244

Dam99. I. Damgård. An error in the mixed adversary protocol by Fitzi, Hirt and Maurer. Available at http://philby.ucsd.edu/cryptolib.html, paper 99-03, 1999. 235

DDWY93. D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *Journal of the ACM*, 40(1):17–47, Jan. 1993. 235, 244, 245

FHM98. M. Fitzi, M. Hirt, and U. Maurer. Trading correctness for privacy in unconditional multi-party computation. In *Advances in Cryptology — CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, 1998. 235, 244, 245

FM98. M. Fitzi and U. Maurer. Efficient Byzantine agreement secure against general adversaries. In *Distributed Computing — DISC '98*, volume 1499 of *Lecture Notes in Computer Science*, Sept. 1998. 242

GHY87. Z. Galil, S. Haber, and M. Yung. Cryptographic computation: Secure fault-tolerant protocols and the public-key model. In *Advances in Cryptology — CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pp. 135–155. Springer-Verlag, 1987. 235, 245

GMW87. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game — a completeness theorem for protocols with honest majority. In *Proc. 19th ACM Symposium on the Theory of Computing (STOC)*, pp. 218–229, 1987. 233

HM97. M. Hirt and U. Maurer. Complete characterization of adversaries tolerable in secure multi-party computation. In *Proc. 16th ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 25–34, Aug. 1997. 233, 234, 235, 237, 240, 241, 244

ISN87. M. Ito, A. Saito, and T. Nishizeki. Secret sharing scheme realizing general access structure. In *Proceedings IEEE Globecom '87*, pp. 99–102. IEEE, 1987. 233

KY. A. Karlin and A. C. Yao. Manuscript. 242

LSP82. L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982. 242

MR98. S. Micali and P. Rogaway. Secure computation: The information theoretic case. Manuscript, 1998. Former version: Secure computation, In *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Note in Computer Science*, pp. 392–404, Springer-Verlag, 1991. 237, 238

RB89. T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. 21st ACM Symposium on the Theory of Computing (STOC)*, pp. 73–85, 1989. 233, 235, 237, 241, 242

SS98. A. Smith and A. Stiglic. Multiparty computation unconditionally secure against Q^2 adversary structures. Manuscript, July 1998. 235, 244

Yao82. A. C. Yao. Protocols for secure computations. In *Proc. 23rd IEEE Symposium on the Foundations of Computer Science (FOCS)*, pp. 160–164. IEEE, 1982. 233