A Complete Treatment of 2-party SFE in the Information-Theoretic Setting with Applications to Long-Term Security

Jörn Müller-Quade[*] and Dominik Raub[†]

April 11, 2008 18:05

Abstract

It is well known that general secure function evaluation (SFE) with information-theoretical (IT) security is infeasible in the secure channels model in presence of a corrupted majority [Cle86, Kil91, Kus92, Kil00, IKLP06, Kat06]. In particular these results extend to and are derived from the 2-party scenario, where any corrupted party is already a corrupted majority. On the other hand [BT07] have recently demonstrated that a wealth of interesting functions can be computed securely even in presence of a corrupted majority, at least if one is willing to sacrifice robustness, thus raising interest in a general description of these functions.

In this work we give a complete combinatorial classification of 2-party functions, by their secure computability under active, semi-honest, passive and quantum adversaries. Our treatment is constructive, in the sense that, if a function is computable in a given setting, then we exhibit a protocol.

We then proceed to apply our results to gain insight into long-term security, where we admit computational assumptions for the duration of a computation, but require information-theoretical security (privacy) once the computation is concluded.

1 Introduction

It is well known that information-theoretically (IT) secure general secure function evaluation (SFE) in presence of a corrupted majority is infeasible in the secure channels model [Cle86, Kil91, Kus92, Kil00, IKLP06, Kat06]. In particular these results extend to and are derived from the 2-party scenario, where any corrupted party is already a corrupted majority. On the other hand [BT07] have recently demonstrated that a wealth of interesting functions can be computed securely even in presence of a corrupted majority, at least if one is willing to sacrifice robustness, thus raising interest in a general description of these securely computable functions.

In this work we attempt a complete combinatorial classification of 2-party functions, by their secure computability under different adversaries. In particular we consider active adversaries that may deviate arbitrarily from the prescribed protocol, passive adversaries[1] that may only observe a correct protocol execution and attempt to extract extra information from the observed interaction, and semi-honest adversaries[2] that are limited in the same way passive adversaries are, but may substitute input and output values in order to gain extra information. We also give some consideration to protocols that may employ a quantum channel and the associated quantum adversaries. We describe the class of 2-party functions $\mathfrak{T}_{2\mathsf{act}}$ that can be IT securely computed in presence of an active adversary, the class $\mathfrak{T}_{2\mathsf{pas}}$ that can be IT securely computed in presence of a passive adversary, and the class $\mathfrak{T}_{2\mathsf{sh}}$, that can be IT securely computed in presence of a semi-honest adversary and which is a

[*]IAKS/EISS, Fakultät für Informatik, Universität Karlsruhe (TH), Germany, muellerq@ira.uka.de

[†]Information Security and Cryptography Research Group, Department of Computer Science, ETH Zurich, Switzerland, raubd@inf.ethz.ch

[1]In the literature our notion of passive is also occasionally referred to as semi-honest.

[2]In the literature our notion of semi-honest is also sometimes referred to as *weakly* semi-honest or *weakly* passive.

superclass of $\mathfrak{T}_{2\mathsf{act}}$ and $\mathfrak{T}_{2\mathsf{pas}}$. Our treatment is constructive, in the sense that, if a function f is computable in a given setting, then our proofs allow to directly derive a secure protocol π computing f.

We then apply our results to long-term (LT) security. Long-term security (LTS) is achieved by protocols which use computational assumptions only during the execution of a protocol and become IT secure afterwards. As such coin flipping protocols and zero-knowledge (ZK) arguments are examples for protocols achieving long-term security. Once these protocols have terminated all computational assumptions can be dropped without impacting security. Both the uniformity of the random coins and the soundness of a ZK argument cannot be invalidated post factum.

We provide a security model for LT security and prove that this notion of security lies strictly between computational security and IT security. More concretely we show that the class of functions $\mathfrak{T}_{\mathsf{lts}}$ computable with LT security in the secure channels model is exactly the class of functions $\mathfrak{T}_{\mathsf{sh}}$ computable with IT security in presence of a semi-honest adversary in the secure channels model.

From our observations for the 2-party case, namely that $\mathfrak{T}_{2\mathsf{act}} \subsetneq \mathfrak{T}_{2\mathsf{sh}} \subsetneq \mathfrak{F}_2$, we can then deduce that the class of LT securely computable functions $\mathfrak{T}_{\mathsf{lts}} = \mathfrak{T}_{\mathsf{sh}}$ in the secure channels model is strictly greater than the class of actively IT securely computable functions $\mathfrak{T}_{\mathsf{act}}$ in the secure channels model, but does not encompass all functions $\mathfrak{F}$. Furthermore we show that the class $\mathfrak{T}_{2\mathsf{qu}}$ of all 2-party functions which can be computed using quantum cryptography is strictly contained in $\mathfrak{T}_{2\mathsf{sh}}$. This gives rise to new impossibility results which complement the well known No-Go Theorems of [May97, LC97].

The basic notion of LT security employed in this work allows for a corrupted party to abort the computation in knowledge of the result. In other words we do not guarantee *quit-fairness*, where quit-fairness is the security property that either all parties learn the result of the computation or nobody does. Indeed from [Cle90] we know that quit-fairness cannot be achieved for general 2-party protocols.

However, computationally (CO) secure protocols can guarantee that only a designated one of the two parties can prematurely abort the protocol after having learned the output [Gol04]. We call this property *security with designated aborter* and show that it can be achieved for all functions in $\mathfrak{T}_{2\mathsf{sh}}$ while preserving long-term security, provided our computational assumption is sufficient for constructing CO secure oblivious transfer (OT) that is also LT secure for one of the participants. As shown in [Gol04] enhanced trap-door one-way permutations are an example of an assumption of sufficient strength. Astonishingly, CO secure OT is used in our construction (where only one party is LT protected), even though OT itself cannot be realized with full LT security. Moreover, we show that a protocol with LT security and designated abort for a specific function in $\mathfrak{T}_{2\mathsf{sh}}$ implies the existence of a protocol for CO secure OT.

1.1 Related Work

Kushilevitz [Kus92] describes the symmetric 2-party functions which can be computed with perfect security in presence of an unbounded passive adversary. In this work we generalize the results of [Kus92] to the asymmetric, statistical case, characterizing the class of functions $\mathfrak{T}_{2\mathsf{pas}}$ which can IT securely be computed in presence of a passive adversary. Our characterization of the classes $\mathfrak{T}_{2\mathsf{sh}}$ and $\mathfrak{T}_{2\mathsf{act}}$, that deal with semi-honest and active adversaries respectively, are in turn built on the characterization of the class $\mathfrak{T}_{2\mathsf{pas}}$. The impossibility result in the quantum case makes use of a result of Kitaev showing the impossibility of quantum coin flipping which is published in [ABDR04]. Some of the ideas presented in this work have already been sketched in [MQ05], however, without proper formalization or proofs. In particular these are the generalization of [Kus92] to the asymmetric, statistical case, connections to LT security and the quantum aspects discussed in this work. The treatment of the class $\mathfrak{T}_{2\mathsf{act}}$ of actively trivial functions, the discussion of designated aborters in the LT setting, and the classification of 2-party functions are to the best of our knowledge original to this work.

Other works that deal with the computability of 2-party functions in the perfect or IT setting are [Kil91, Kil00, BMM99, KMQ08]. However, these papers focus mostly on reducibility and completeness, while we are more interested in computability in the plain setting and implications for LT security.

Everlasting security from temporary assumptions has been investigated in cryptographic research for some time. It was shown that a bound on the memory available to the adversary allows to implement key exchange protocols [CM97] and protocols for oblivious transfer [CCM02] which remain secure even if the memory bound held only during the execution of the protocol. This idea has further been persued to achieve everlasting security from a network of distributed servers providing randomness [Rab03]. In [DM04] it was shown that using a computationally secure key exchange in the bounded storage model need not yield everlasting security. For some time general quantum cryptographic protocols were sought which obtain everlasting security from a temporary assumption. Such protocols are now generally accepted to be impossible [BCMS99]. Additional assumptions, like a temporary bound on the quantum memory can again provide everlasting security for secure computations [DFSS05].

In this paper we investigate the power of temporary computational assumptions in the standard model. This is along the lines of [MQU07]. However in [MQU07] strong composability requirements are imposed under which little is possible without additional setup assumptions, like the temporary availability of secure hardware.

2 Security Models

We use a simulation based stand-alone model of security (as opposed to a universally composable model) with synchronous message passing where parties are connected via secure channels as for instance described in [Gol04]. Security is defined by comparing, by means of a distinguisher D, a real protocol π that interacts with an adversary E with an ideal functionality I that interacts with a simulator S. We generally demand that for all adversaries $\mathsf{E} \in \mathcal{E}$ there are simulators $\mathsf{S}(\mathsf{E}) \in \mathcal{S}$ such that for all distinguishers $\mathsf{D} \in \mathcal{D}$ the advantage $\Delta_D(\mathsf{S}(\mathsf{E}) \circ I, \mathsf{E} \circ \pi \circ R) = \Delta(\mathsf{D}(\mathsf{S}(\mathsf{E}) \circ I), \mathsf{D}(\mathsf{E} \circ \pi \circ R)) < \epsilon(\kappa)$ where Δ denotes statistical distance, $\mathcal{E}, \mathcal{S}, \mathcal{D}$ denote classes of algorithms and distinguishers make binary output. A more detailed treatment of the security model can be found in Appendix A.

We consider both computational (CO) security, where distinguishers, adversaries and simulators must be efficient[3] ($\mathcal{D}, \mathcal{E}, \mathcal{S} \subseteq \mathsf{Poly}$) and information-theoretic (IT) security where distinguishers, adversaries and simulators are arbitrary unbounded algorithms ($\mathcal{D}, \mathcal{E}, \mathcal{S} \subseteq \mathsf{Algo}$). Furthermore, we investigate long-term (LT) security where adversaries and simulators must be efficient ($\mathcal{E}, \mathcal{S} \subseteq \mathsf{Poly}$) but distinguishers may be unbounded ($\mathcal{D} \subseteq \mathsf{Algo}$), formalizing that we expect computational assumptions to hold only for the duration of the protocol. In all three cases the advantage $\epsilon(\kappa)$ is chosen to be negligible in the security parameter κ. If in the IT case we fix an advantage of $\epsilon = 0$ we arrive at perfect (PF) security. We may vary the IT and PF cases by demanding efficient simulators (ITE, PFE).

We refine these security paradigms further by defining adversarial models, i.e. restrictions that we can impose on the adversaries and simulators for any of the above paradigms. We discuss active (ACT) adversaries, where adversaries and simulators are not restricted further; semi-honest (SH) adversaries, where adversaries in the class $\mathcal{E}_{\mathsf{sh}}$ are restricted to generate messages according to the prescribed protocol π with the inputs provided by the distinguisher D, and simulators are not restricted further; and passive (PA) adversaries, where adversaries in the class $\mathcal{E}_{\mathsf{pas}} = \mathcal{E}_{\mathsf{sh}}$ are restricted to generate messages according to the prescribed protocol π with the inputs provided by the distinguisher D, and simulators in the class $\mathcal{S}_{\mathsf{pas}}$ are restricted to forward the inputs provided by the distinguisher D to the ideal functionality I.

Security paradigms and adverserial models as defined above are combined by intersecting their defining sets, i.e. semi-honest IT security is described by $\mathcal{D}^{\mathsf{IT}}_{\mathsf{sh}} = \mathcal{D}^{\mathsf{IT}}$, $\mathcal{S}^{\mathsf{IT}}_{\mathsf{sh}} = \mathcal{S}^{\mathsf{IT}} \cap \mathcal{S}_{\mathsf{sh}}$, $\mathcal{E}^{\mathsf{IT}}_{\mathsf{sh}} = \mathcal{E}^{\mathsf{IT}} \cap \mathcal{E}_{\mathsf{sh}}$, $\epsilon(\kappa) < \mathsf{negl}$ and denoted $\pi \succcurlyeq^{\mathsf{IT}}_{\mathsf{sh}} I$.

We can now formalize security requirements under each of the definitions above by providing an appropriate ideal system.

[3]By efficient we mean polynomially bounded in the security parameter κ.

Let $\mathfrak{P} = \{P_1, \ldots, P_n\}$ be a set of n parties[4] and let $\mathfrak{F}_n$ designate the set of all n-party functions

$$f : \begin{cases} \mathcal{X}_1 \times \ldots \times \mathcal{X}_n \to \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_n \\ (x_1, \ldots, x_n) \mapsto f(x_1, \ldots, x_n) = (y_1, \ldots, y_n). \end{cases} \tag{1}$$

where the input and output sets $\mathcal{X}_1, \ldots, \mathcal{X}_n, \mathcal{Y}_1, \ldots, \mathcal{Y}_n \subset \mathbb{N}$ for the n parties are finite. Furthermore, let $\mathfrak{F} := \bigcup_{n \geq 1} \mathfrak{F}_n$ denote the set of all multi-party functions. Now let $f \in \mathfrak{F}_n$ be a concrete function[5] and let $\mathfrak{C} \subset \mathfrak{P}$ be a set of corrupted players.

Definition 2.1 (Security (with Robustness))**.** We define an ideal functionality I_f that formalizes computing f securely (including robustness) as follows: First $I_f(\mathfrak{C})$ takes inputs $x_i \in \mathcal{X}_i$ from each party P_i. If no valid input is received from P_i then a default value x_i^{def} for x_i is assumed. Now $I_f(\mathfrak{C})$ computes $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$ and outputs y_i to P_i for all $P_i \in \mathfrak{P}$. $\Diamond$

Unfortunately, in protocols tolerating a majority of dishonest parties robustness and even quit-fairness cannot generally be guaranteed [Cle86, Cle90, IKLP06, Kat06]. In other words, an adversary can possibly abort the protocol even after obtaining (part of) the protocol result, while preventing the honest parties from learning their result. Although such a scenario is clearly less desirable then fair or even robust computation, we are still interested in security in presence dishonest majorities. Thus, we devise an ideal system that explicitly allows for this type of behavior:

Definition 2.2 (Security with Abort)**.** We define an ideal functionality $I_f^{\mathsf{ab}}(\mathfrak{C})$ that formalizes computing a function f securely with abort as follows: First $I_f^{\mathsf{ab}}(\mathfrak{C})$ takes inputs $x_i \in \mathcal{X}_i$ from each party P_i. If no valid input x_i is received from P_i then a default value x_i^{def} for x_i is assumed. Now $I_f^{\mathsf{ab}}(\mathfrak{C})$ computes $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$ and outputs y_i for $P_i \in \mathfrak{C}$ to the ideal adversary (simulator) S. The ideal functionality $I_f^{\mathsf{ab}}(\mathfrak{C})$ then awaits an input $o \in \{0, 1\}$ from the the simulator S. On $o = 1$ the functionality $I_f^{\mathsf{ab}}(\mathfrak{C})$ distributes y_i to all parties P_i, on $o = 0$ it sends $\perp$ to all parties P_i and terminates. $\Diamond$

As mentioned above, in protocols tolerating up to $n - 1$ out of n corrupted parties successful termination cannot generally be guaranteed. A single corrupted party can possibly abort the protocol. However, it is also known that CO secure SFE protocols can be designed in such a way that only a single party, to which we refer as the designated aborter, can abort the protocol *and* (computationally) learn anything about the result of the computation. In other words, only the designated aborter can abort the protocol conditional on his output. Aborts from other players are no different from refusal to participate in the protocol. This notion of *computational security with designated aborter* is of practical interest, because we may in an application setting have a party which is not fully trusted but can be relied upon not to abort the protocol. We hence formalize this notion in the following definition:

Definition 2.3 (Security with Designated Aborter (DA))**.** We define an ideal functionality $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ that formalizes securely computing a function f with designated aborter P_j as follows: First $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ takes inputs $x_i \in \mathcal{X}_i$ from each party P_i. If no valid input is received from P_i then a default value x_i^{def} for x_i is assumed. Now $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ computes $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$. If $P_j \in \mathfrak{C}$ the functionality $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ outputs y_i for $P_i \in \mathfrak{C}$ to the ideal adversary (simulator) S. Else it only outputs y_j to P_j. $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ then awaits an input $o \in \{0, 1\}$ from the designated aborter P_j. On $o = 1$ the ideal system $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ distributes y_i to all parties P_i, on $o = 0$ it sends $\perp$ to all parties and terminates.

We generally set $j = 1$ and drop it from the notation. $\Diamond$

[4]Throughout this work, we will mostly consider the case $n = 2$ and $\mathfrak{P} = \{P_A, P_B\}$.

[5]In this work we take the function f to be independent of the security parameter κ. This is the natural and most relevant case for applications. Our proofs would however still hold for a family of functions f_κ, where the input domain grows at most polynomially fast in the security parameter κ.

Naturally we investigate how the notions of LT security and computational security with designated aborter combine. To this end we define

Definition 2.4 (Long Term Security with Designated Aborter (LTS-DA))**.** A protocol π computes a function f with long-term security with designated aborter (LTS-DA) if it computes f with CO security with designated aborter and if the protocol π' where the reply of an honest P_1 after receiving its output is fixed to ok computes f with LT security with abort. $\Diamond$

Note that in LTS-DA long-term protection of secrets is only guaranteed for protocols which terminate. In case of an abort an adversary may, in the long term, learn intermediate results (by breaking the computational assumption).

3 The Class of Passively Trivial Functions

Kushilevitz [Kus92] characterizes the symmetric 2-party functions that can be perfectly securely computed in presence of a passive adversary. In this section, we generalize this result in two ways: First we admit asymmetric functions, where the participants A and B may obtain different outputs. Second we consider passive IT security, so we admit a statistically small (negligible) chance that security fails.

As already pointed out in the introduction, we discuss the secure evaluation of a function f which is independent of the security parameter κ. This is the natural and most relevant case for applications. However, our proofs still hold for a family of functions f_κ, where the input domain $\mathcal{X}_A \times \mathcal{X}_B$ grows at most polynomially fast in the security parameter κ. For function families f_κ, where the input domain $\mathcal{X}_A \times \mathcal{X}_B$ grows superpolynomially fast in the security parameter κ, we cannot make a statement, since, as pointed out in [Kus92], such functions may require superpolynomially many rounds to compute, invalidating the inductive arguments over negligible quantities employed in our proofs.

We begin by defining the class of 2-party functions that can securely be computed in presence of a passive adversary:

Definition 3.1 (Passively Trivial)**.** The class of 2-party functions $f \in \mathfrak{F}_2$ for which an efficient protocol $\pi \in \mathsf{Poly}$ exists, implementing I_f with passive IT security is called the class of *passively trivial* functions $\mathfrak{T}_{2\mathsf{pas}}$. $\Diamond$

Before we describe the class of passively trivial functions $\mathfrak{T}_{2\mathsf{pas}}$ combinatorially, we note that locally computable functions are clearly passively trivial. For a function $f : \mathcal{X}_A \times \mathcal{X}_B \to \mathcal{Y}_A \times \mathcal{Y}_B$ and a projection $\pi_i : \mathcal{Y}_A \times \mathcal{Y}_B \to \mathcal{Y}_i$ we write f_i to denote $\pi_i \circ f$ ($i \in \{A, B\}$). Now we can characterize the class of locally computable functions:

Lemma 3.2 (Local Computability)**.** *A 2-party function $f \in \mathfrak{F}_2$ is* locally computable*, without interaction between the parties A and B, if and only if the results of each party are independent of the inputs of the other party. In other words $f_B|_{\mathcal{X}_A \times \{x_B\}}$ is constant for any $x_B \in \mathcal{X}_B$ and $f_A|_{\{x_A\} \times \mathcal{X}_B}$ is constant for any $x_A \in \mathcal{X}_A$. We call the set of locally computable functions $\mathfrak{T}_{2\mathsf{loc}}$.*

The proof is trivial and left to the reader. The following combinatorial characterization of passively trivial functions is then a generalization of the one by [Kus92] to the asymmetric setting.

Definition 3.3 (Passively Decomposable)**.** A 2-party function $f \in \mathfrak{F}_2$ is called *passively decomposable* if for any restriction $f|_{\widetilde{\mathcal{X}}_A \times \widetilde{\mathcal{X}}_B}$ of f to subsets $\widetilde{\mathcal{X}}_A \subseteq \mathcal{X}_A$, $\widetilde{\mathcal{X}}_B \subseteq \mathcal{X}_B$ we have

1. $f|_{\widetilde{\mathcal{X}}_A \times \widetilde{\mathcal{X}}_B} \in \mathfrak{T}_{2\mathsf{loc}}$ locally computable or
2. there is a partition (K-cut) into non-empty sets $\mathcal{X}'_A \cup \mathcal{X}''_A = \widetilde{\mathcal{X}}_A$ such that for any $x_B \in \widetilde{\mathcal{X}}_B$ we have $f_B(\mathcal{X}'_A, x_B) \cap f_B(\mathcal{X}''_A, x_B) = \emptyset$ or

3. there is a partition (K-cut) into non-empty sets $\mathcal{X}'_B \cup \mathcal{X}''_B = \widetilde{\mathcal{X}}_B$ such that for any $x_A \in \widetilde{\mathcal{X}}_A$ we have $f_A(x_A, \mathcal{X}'_B) \cap f_A(x_A, \mathcal{X}''_B) = \emptyset$.

The set of passively decomposable functions will be called $\mathfrak{T}'_{\mathsf{2pas}}$. $\Diamond$

Now we state that our combinatorial description indeed characterizes the passively trivial functions:

Lemma 3.4 (Passive Triviality)**.** *A 2-party function $f \in \mathfrak{F}_2$ is passively trivial if and only if it is passively decomposable. In short $\mathfrak{T}_{\mathsf{2pas}} = \mathfrak{T}'_{\mathsf{2pas}}$. Furthermore, any function $f \in \mathfrak{T}_{\mathsf{2pas}}$ can efficiently be computed with perfect security.*

The proof of this lemma can be found in Appendix B. It generalizes the proof of [Kus92] to the asymmetric, statistical setting. Note that the first part of the proof is actually constructive in the sense that it inductively describes a passively PF secure protocol π_f for the function $f \in \mathfrak{T}_{\mathsf{2pas}} = \mathfrak{T}'_{\mathsf{2pas}}$. This protocol π_f is essentially the same as in [Kus92]:

Let A and B have inputs $a \in \mathcal{X}_A$ and $b \in \mathcal{X}_B$ respectively. Wlog assume that there is a decomposition $\mathcal{X}_B = \mathcal{X}_B^{(1)} \dot{\cup} \mathcal{X}_B^{(2)}$ as described in Def. 3.3 (else interchange A and B). Then B determines the message $m_1 \in \{1, 2\}$ such that B's input $b \in \mathcal{X}_B^{(m_1)}$ and sends m_1 to A. A and B then restrict the function f to $f|_{\mathcal{X}_A \times \mathcal{X}_B^{(m_1)}}$ and proceed with a partition for $f|_{\mathcal{X}_A \times \mathcal{X}_B^{(m_1)}}$ in the same fashion. The process is iterated until the parties arrive at a locally computable restriction of f, at which point they compute the output locally.

The round complexity of π_f can be improved by using the finest possible decomposition, for instance $\mathcal{X}_B = \mathcal{X}_B^{(1)} \dot{\cup} \cdots \dot{\cup} \mathcal{X}_B^{(u)}$, where the $\mathcal{X}_B^{(i)}$ fulfill the criterion $\forall x_A \in \mathcal{X}_A,\ i \neq j\ :\ f_A(x_A, \mathcal{X}_B^{(i)}) \cap f_A(x_A, \mathcal{X}_B^{(j)}) = \emptyset$. [Kus92] shows that the resulting protocol π_f achieves the optimal round complexity for passive PF security.

4 The Class of Semi-Honestly Trivial Functions

Consider a LTS scenario, where during a protocol run we can rely on computational assumptions. Then, in presence of an active adversary, we can force honest behavior using zero-knowledge arguments and unconditionally hiding bit-commitments. What we cannot prevent, however, is corrupted parties exchanging their inputs for different ones. We call such behavior, where corrupted parties may change their input before beginning the computation and try to obtain extra information but adhere to the protocol otherwise *semi-honest*.

A priori, this suggests stronger adversaries than the strictly passive adversary class $\mathcal{E}_{\mathsf{sh}} = \mathcal{E}_{\mathsf{pas}}$ from the definition of semi-honest security in Section 2, namely adversaries E that can exchange inputs provided by the distinguisher D.

We show that such an apparently stronger definition of security is still equivalent to the definition of semi-honest security in Section 2: Note that for the distinguisher classes $\mathcal{D}$ under consideration in this work, for any distinguisher $\mathsf{D} \in \mathcal{D}$ we find a distinguisher $\mathsf{D}' = \mathsf{D} \circ \sigma \in \mathcal{D}$ that incorporates the input substitution σ of $\mathsf{E} = \mathsf{E}' \circ \sigma$. In other words we can find a passive adversary $\mathsf{E}' \in \mathcal{E}_{\mathsf{sh}}$ and a distinguisher D' that yield the same advantage as E and D. So the definition of semi-honest security from Section 2 is equivalent to the modified definition.

Next, we define the set of semi-honestly trivial functions, that are computable in this setting:

Definition 4.1 (Semi-Honestly Trivial)**.** The class of 2-party functions $f \in \mathfrak{F}_2$ for which an efficient protocol $\pi \in \mathsf{Poly}$ exists, implementing I_f with semi-honest IT security is called the class of *semi-honestly trivial* functions $\mathfrak{T}_{\mathsf{2sh}}$. $\Diamond$

As the outputs for A and B are different, some inputs of A may be completely indistinguishable for B, while one of these inputs may yield more information for A (and vice versa). More precisely, an input x of one party, say A, to a function f is said to *dominate* another input x' if the two inputs are indistinguishable for the other party B and every two inputs of the other party B which can be distinguished by A by entering x' can also be distinguished by entering x (also see [KMQ08]).

The dominated input x' gives less information than x and is not useful for a corrupted A. Along the same lines an ideal adversary (simulator) can always use the dominating input x for simulation. As such the input x' is redundant. It makes no difference in terms of security, and we can eliminate it from the function f under consideration. This procedure yields a *redundancy-free version* $\hat{f}$ of f with new, smaller, dominating input sets $\hat{\mathcal{X}}_A$ and $\hat{\mathcal{X}}_B$. Precise definitions of dominance and redundancy-freeness can be found in Appendix C or [KMQ08].

A function f and its redundancy-free version $\hat{f}$ are mutually reducible with perfect security and using local transformations only, i.e. without additional communication resources:

Lemma 4.2 (Local Mutual Reducibility)**.** *The function f and its redundancy-free version $\hat{f}$ are locally mutually reducible with perfect security ($\mathcal{D}, \mathcal{E} \subseteq \mathsf{Algo}$) and efficient simulator ($\mathcal{S} \supseteq \mathsf{Poly}$), even in presence of an active adversary, i.e. $I_f \succcurlyeq^{\mathsf{PFE}}_{\mathsf{act}} I_{\hat{f}} \succcurlyeq^{\mathsf{PFE}}_{\mathsf{act}} I_f$.*

The proof of this lemma can be found in Appendix D. We are now ready to characterize the class of semi-honestly trivial functions $\mathfrak{T}_{2\mathsf{sh}}$:

Lemma 4.3 (Semi-Honest Triviality)**.** *A 2-party function $f \in \mathfrak{F}_2$ is semi-honestly trivial ($f \in \mathfrak{T}_{2\mathsf{sh}}$) iff the redundancy-free version $\hat{f}$ of f is passively trivial ($\hat{f} \in \mathfrak{T}_{2\mathsf{pas}}$). Furthermore any function $f \in \mathfrak{T}_{2\mathsf{sh}}$ can be computed efficiently with perfect security.*

The proof of this lemma can be found in Appendix E. The functions $f^{(5)}$ and $f^{(6)}$ in Fig. 1 are examples of *not* semi-honestly trivial functions taken from [Kus92]. The function $f^{(6)}$ is of particular interest as it is of strictly less cryptographic strength as oblivious transfer.

5 The Class of Actively Trivial Functions

In this section we describe the class of all 2-party functions which can securely be computed in presence of an unlimited active adversary. We will call this class of functions the class $\mathfrak{T}_{2\mathsf{act}}$ of actively trivial functions. Our combinatorial classification below implies that $\mathfrak{T}_{2\mathsf{act}}$ is strictly contained in $\mathfrak{T}_{2\mathsf{sh}}$ and hence the notion of LT security lies strictly between IT security and CO security. Interestingly there are some useful functions in the class $\mathfrak{T}_{2\mathsf{act}}$, e.g. $f^{(7)}$ in Fig. 1 which is a formalization of a Dutch flower auction, where the price is lowered in every round until a party decides to buy.

Definition 5.1 (Actively Trivial)**.** The class of 2-party functions $f \in \mathfrak{F}_2$ for which an efficient protocol $\pi \in \mathsf{Poly}$ exists, implementing I_f with active IT security is called the class of *actively trivial* functions $\mathfrak{T}_{2\mathsf{act}}$. $\Diamond$

We then give a combinatorial characterization of actively trivial functions:

Definition 5.2 (Actively Decomposable)**.** A 2-party function $f \in \mathfrak{F}_2$ is called *actively decomposable*, iff $\hat{f} \in \widehat{\mathfrak{T}_{2\mathsf{act}}}$. A function $f \in \mathfrak{F}_2$ is in $\widehat{\mathfrak{T}_{2\mathsf{act}}}$ if one of the following holds

1. $f \in \mathfrak{T}_{2\mathsf{loc}}$ is locally computable;
2. there is a partition (T-cut) of $\mathcal{X}_B = \mathcal{X}'_B \cup \mathcal{X}''_B$ such that $f\mid_{\mathcal{X}_A \times \mathcal{X}'_B}, f\mid_{\mathcal{X}_A \times \mathcal{X}''_B} \in \widehat{\mathfrak{T}_{2\mathsf{act}}}$ and
$$\begin{aligned} \forall\, x'_A, x''_A \in \mathcal{X}_A\ \exists x_A \in \mathcal{X}_A \forall\, x'_B \in \mathcal{X}'_B, x''_B \in \mathcal{X}''_B\ :\quad & f_A(x'_A, x'_B) \neq f_A(x'_A, x''_B) \quad \wedge \\ & f_B(x'_A, x'_B) = f_B(x_A, x'_B) \wedge f_B(x''_A, x''_B) = f_B(x_A, x''_B); \end{aligned}$$
3. the above with parties A and B interchanged.

The set of passively decomposable functions f will be called $\mathfrak{T}'_{\mathsf{2act}}$. $\diamondsuit$

The functions $f^{(7)}$ and $f^{(8)}$ in Fig. 1 are examples of actively trivial functions. Especially compare $f^{(8)}$ with $f^{(2)} \in \mathfrak{T}_{\mathsf{2sh}}$ which is *not* actively trivial. The protocol used for computing actively trivial functions is the same as in the passive case. The lines in the tables for $f^{(7)}$ and $f^{(8)}$ represent messages which are to be sent in the protocol. Each message informs the other party on which side of the line the result lies. Intuitively speaking the table becomes smaller for every message sent until the result is obtained. For both functions the party B whose input fixes the column must start the protocol sending the message corresponding to the longest line in the table.

We now state that active decomposability actually characterizes the actively trivial functions.

Theorem 5.3 (Active Triviality)**.** *A 2-party function f is actively trivial if and only if it is actively decomposable. In short $\mathfrak{T}_{\mathsf{2act}} = \mathfrak{T}'_{\mathsf{2act}}$. Furthermore any function $f \in \mathfrak{T}_{\mathsf{2act}}$ can be computed efficiently with perfect security.*

The proof of this theorem can be found in Appendix F.

6 Quantum Protocols

In this section we will relate the class $\mathfrak{T}_{\mathsf{2sh}}$ of SH trivial two argument functions with the class of two argument functions achievable by quantum cryptography in presence of an active adversary. A similar result has been obtained by Louis Salvail, but is not published yet. For this result we have to adapt our model of security to the quantum case. All machines except for the distinguisher D will be quantum machines able to exchange quantum messages. The adversary will be unbounded. Furthermore, all inputs and outputs must be classical and the distinguisher must try to distinguish the real and the ideal model on basis of this classical information.

Let $\mathfrak{T}_{\mathsf{2qu}}$ denote the set of functions $f \in \mathfrak{F}_2$ which can, with the help of a quantum channel, securely and effciently be computed in presence of an unbounded adversary. Then the following result holds.

Theorem 6.1. *The class $\mathfrak{T}_{\mathsf{2qu}}$ of quantum trivial functions is strictly contained in the class of SH trivial functions $\mathfrak{T}_{\mathsf{2sh}}$.*

A proof of this theorem is sketched in Appendix G. The strict inclusion $\mathfrak{T}_{\mathsf{2qu}} \subsetneq \mathfrak{T}_{\mathsf{2sh}}$ gives rise to new impossibility results. For instance, the function $f^{(7)} \notin \mathfrak{T}_{\mathsf{2sh}}$ in Fig. 1 cannot be computed by means of quantum cryptography. An interesting still open question is the power of temporary computational assumptions together with a quantum channel. It is known that this does not suffice to securely implement any function which could in turn be used to implement an IT secure bit commitment. However a secure implementation of the function $f^{(7)}$ in Fig. 1 is not precluded by this impossibility result.

7 Long-Term Security

In this section we characterize the class $\mathfrak{T}_{\mathsf{2lts}}$ of 2-party functions that can be computed with long-term security in presence of an active adversary. So we ask which functions $f \in \mathfrak{F}_2$ can be securely computed, if we are willing to make computational assumptions, but only for the duration of the protocol interaction. Once the protocol has terminated we demand IT security.

We shall see that $\mathfrak{T}_{\mathsf{2act}} \subsetneq \mathfrak{T}_{\mathsf{2lts}} \subsetneq \mathfrak{F}_2$. Indeed $\mathfrak{T}_{\mathsf{2lts}} = \mathfrak{T}_{\mathsf{2sh}}$, as unconditionally hiding bit-commitments and perfect zero-knowledge arguments allow to turn protocols which are secure against unbounded semi-honest attackers into LT secure protocols.

Theorem 7.1. *Assuming the existence of one-way functions (OWF), the class $\mathfrak{T}_{\mathsf{2lts}}$ of 2-party functions f which can be computed with LT security in presence of an active adversary, i.e. $f \succeq^{\mathsf{LT}}_{\mathsf{act}} I^{\mathsf{ab}}_{f}$, is exactly the class of functions $\mathfrak{T}_{\mathsf{2sh}}$ that can be computed with IT security in presence of a semi-honest adversary.*

A proof of this theorem is sketched in Appendix H. Also, note that this theorem is easily generalized to the multi-party setting.

Against active adversaries robustness or even fairness can generally only be guaranteed for functions $f \in \mathfrak{T}_{\mathsf{2act}}$ [Cle86, Cle90]. Thus we only demand that I_f^{ab} be implemented, which allows a corrupted party to abort the protocol after obtaining output and before the honest parties can generate output.

7.1 Long Term Security with designated Aborter

As mentioned above we cannot generally guarantee robustness or even fairness for a LT secure protocol π_f computing $f \in \mathfrak{T}_{\mathsf{2lts}}$. However, under stronger computational assumptions, we can guarantee that only a specific designated party can abort the protocol after obtaining output and before the honest parties can generate output. This may be of practical relevance where a specific party is not trusted, but can be relied upon not to abort the protocol. For instance a party may have a vested interest in the successful termination of the protocol regardless of the outcome. One may think of an auctioneer that gets paid only if the auction terminates successfully. Or a party may act in an official capacity and cannot abort the protocol for legal reasons.

We will show that stronger guarantees of this type are obtainable if and only if the underlying computational assumption allows for an oblivious transfer (OT) protocol which is LT secure against one of the participants. Enhanced trapdoor one-way permutations are an example of such an assumption [Gol04]. It is generally believed that OT is *not* implied by OWFs, meaning that LT security with designated aborter requires strictly stronger assumptions than plain LT security.

Lemma 7.2. *Any SH trivial 2-party function $f \in \mathfrak{T}_{\mathsf{2sh}} = \mathfrak{T}_{\mathsf{2lts}}$ can be computed using a protocol π which is LTS-DA iff computational oblivious transfer LT-secure against one party (CO-OT+) exists.*

A proof of this lemma is sketched in Appendix I.

8 Classification of 2-party Functions

Combining the results from this work and from [KMQ08], we arrive at a complete combinatorial classification of the 2-party functions $\mathfrak{F}_2$.

We first define an equivalence relation *renaming* $\equiv$ on $\mathfrak{F}_2$ by $f^{(1)} \equiv f^{(2)}$ iff $f^{(2)}$ is obtained from $f^{(1)}$ by locally renaming input and output values. A formal definition can be found in [KMQ08] or Appendix J. It is easy to see that renamings are locally mutually reducible under all security paradigms considered in this work. In particular $f^{(1)} \equiv f^{(2)}$ implies $I_{f^{(1)}} \succeq^{\mathsf{PFE}}_{\mathsf{act}} I_{f^{(2)}} \succeq^{\mathsf{PFE}}_{\mathsf{act}} I_{f^{(1)}}$ and $I_{f^{(1)}} \succeq^{\mathsf{PFE}}_{\mathsf{pas}} I_{f^{(2)}} \succeq^{\mathsf{PFE}}_{\mathsf{pas}} I_{f^{(1)}}$.

Next we define an equivalence relation *matching* on $\mathfrak{F}_2/\equiv$ (and thereby on $\mathfrak{F}_2$) by isolating inputs that lead to identical behavior and regarding functions as matching if, after eliminating such trivially redundant inputs, they are renamings:

Definition 8.1. Given a 2-party function $f \in \mathfrak{F}_2$ we say x_A *matches* x'_A for inputs $x_A, x'_A \in \mathcal{X}_A$, iff x_A dominates x'_A and x'_A dominates x_A. The matching relation is an equivalence relation on $\mathcal{X}_A$. By $\bar{\mathcal{X}}_A$ we designate a set of representatives. $\bar{\mathcal{X}}_B$ is defined analogously.

We then call $\bar{f} := f|_{\bar{\mathcal{X}}_A \times \bar{\mathcal{X}}_B}$ the *weakly redundancy-free version* of f and for $f^{(1)}, f^{(2)} \in \mathfrak{F}_2$ we write $f^{(1)} \cong f^{(2)}$ if $\bar{f}^{(1)} \equiv \bar{f}^{(2)}$ Furthermore for $x_A \in \mathcal{X}_A$ and $x_B \in \mathcal{X}_B$ let $\bar{x}_A \in \bar{\mathcal{X}}_A$ and $\bar{x}_B \in \bar{\mathcal{X}}_B$ be the (unique) elements that match x_A respectively x_B. $\diamondsuit$

Like the redundancy-free version $\hat{f}$ of f, the weakly redundancy-free version $\bar{f}$ of f is well defined up to renaming. Before we can state the actual classification, we have to reiterate another result of [KMQ08]:

Theorem 8.2 (Complete Functions [KMQ08])**.** *The classes $\mathfrak{C}_{\mathsf{2act}}$, $\mathfrak{C}_{\mathsf{2sh}}$ and $\mathfrak{C}_{\mathsf{2pas}}$ of actively, semi-honestly, and passively complete 2-party functions are the classes of functions $f \in \mathfrak{F}_2$ to which all other 2-party functions can be securely reduced in presence of an active, semi-honest or passive adversary respectively.*

The class $\mathfrak{C}_{\mathsf{2pas}}$ consists of exactly the functions $f \in \mathfrak{F}_2$ where

$$\begin{aligned}\exists\, a_1, a_2 \in \mathcal{X}_A,\ b_1, b_2 \in \mathcal{X}_B : \qquad & f_A(a_1, b_1) = f_A(a_1, b_2)\ \wedge f_B(a_1, b_1) = f_B(a_2, b_1)\ \wedge \\ & (\ f_A(a_2, b_1) \neq f_A(a_2, b_2)\ \vee f_B(a_1, b_2) \neq f_B(a_2, b_2)\ ).\end{aligned}$$

We refer to this combinatorial structure as minimal OT.

The classes $\mathfrak{C}_{\mathsf{2act}} = \mathfrak{C}_{\mathsf{2sh}}$ consist of exactly the functions $f \in \mathfrak{F}_2$ where $\hat{f} \in \mathfrak{C}_{\mathsf{2pas}}$.

Note that $f \in \mathfrak{C}_{\mathsf{2pas}}$ iff $f \in \mathfrak{C}_{\mathsf{2act}}$ or $\hat{f} \not\equiv \bar{f}$. This is clear from Kraschewki's result as stated above and from the observation that $\hat{f} \not\equiv \bar{f}$ implies a minimal OT. We then arrive at the following

Theorem 8.3 (Classification). *The class of 2-party functions is a disjoint union of three sets $\mathfrak{F}_2 = \mathfrak{C}_{\mathsf{2act}} \cup \mathfrak{T}_{\mathsf{2act}} \cup \mathfrak{F}^{\mathsf{nct}}_{\mathsf{2act}}$ or $\mathfrak{F}_2 = \mathfrak{C}_{\mathsf{2sh}} \cup \mathfrak{T}_{\mathsf{2sh}} \cup \mathfrak{F}^{\mathsf{nct}}_{\mathsf{2sh}}$ or $\mathfrak{F}_2 = \mathfrak{C}_{\mathsf{2pas}} \cup \mathfrak{T}_{\mathsf{2pas}} \cup \mathfrak{F}^{\mathsf{nct}}_{\mathsf{2pas}}$ where* nct *stand for "neither complete nor trivial". Now*

$$\emptyset \neq \mathfrak{T}_{\mathsf{2act}}, \mathfrak{T}_{\mathsf{2pas}} \subsetneq \mathfrak{T}_{\mathsf{2act}} \cup \mathfrak{T}_{\mathsf{2pas}} \subsetneq \mathfrak{T}_{\mathsf{2sh}} \tag{2}$$

$$\emptyset \neq \mathfrak{F}^{\mathsf{nct}}_{\mathsf{2pas}} \subsetneq \mathfrak{F}^{\mathsf{nct}}_{\mathsf{2sh}} \subsetneq \mathfrak{F}^{\mathsf{nct}}_{\mathsf{2act}} \tag{3}$$

$$\emptyset \neq \mathfrak{C}_{\mathsf{2act}} = \mathfrak{C}_{\mathsf{2sh}} \subsetneq \mathfrak{C}_{\mathsf{2pas}} \tag{4}$$

The above results are directly derived from the combinatorial descriptions of the function classes that can be found in the preceding sections and, as far as complete functions are concerned, in [KMQ08]. Additional details and examples can be found in Appendix J.

9 Conclusions

We gave combinatorial characterizations of the classes $\mathfrak{T}_{\mathsf{2act}}$, $\mathfrak{T}_{\mathsf{2sh}}$, $\mathfrak{T}_{\mathsf{2pas}}$ of 2-party functions that can be computed with IT security in presence of an active, semi-honest and passive adversary respectively. These characterizations are constructive in that they directly imply a protocol computing a computable function.

We then investigated the class $\mathfrak{T}_{\mathsf{2lts}}$ of 2-party functions computable with long-term security. These are the functions that can be computed with information theoretic security, provided the adversary is computationally bounded *during* the execution of the protocol. That is, we rely on computational assumptions, but only for the duration of the protocol execution. Thereafter, a failure of the computational assumptions may not compromise security. As such LT security is a practically very appealing notion of security, since we are fairly confident that our computational assumptions hold at present, but may have to deal with quantum computers, new factoring algorithms and better hardware in the future. IT secure protocols do address these issues of course, but they generally only offer security if a majority of the participants is honest. Security in absence of a honest majority requires computational assumptions and if we are unwilling to rely on computational assumptions for our security indefinitely, long-term security is the next best solution.

We showed that the class $\mathfrak{T}_{\mathsf{2lts}}$ of functions computable with LT security is equal to the class $\mathfrak{T}_{\mathsf{2sh}}$ of functions computable with IT security in presence of a semi-honest adversary (This result generalizes straightforwardly to the multi-party case.). The equality $\mathfrak{T}_{\mathsf{2lts}} = \mathfrak{T}_{\mathsf{2sh}}$ provides us with a combinatorial characterization of the functions computable with LT security. Furthermore, from $\mathfrak{T}_{\mathsf{2act}} \subsetneq \mathfrak{T}_{\mathsf{2sh}} = \mathfrak{T}_{\mathsf{2lts}} \subsetneq \mathfrak{F}_2$ we find that temporary computational assumptions are indeed useful, as they allow us to compute strictly more functions under an active adversary than would be possible with pure IT security.

Finally, we showed that the class $\mathfrak{T}_{\mathsf{2qu}}$ of two argument functions which can be implemented with quantum cryptography is strictly contained in $\mathfrak{T}_{\mathsf{2sh}}$. This looks like quantum cryptography is not of much use besides establishing a secure channel, but this is jumping to conclusions. Quantum cryptography can solve classically impossible problems in different models of security, like achieving a certain robustness to abort in a model with guaranteed message delivery or implementing a key exchange which is deniable.

$f^{(1)}$	0	1
0	0/0	0/0
1	0/0	1/0

$f^{(2)}$	0	1
0	0	1
1	0	2
2	3	2

$f^{(3)}$	0	1	2
0	0/0	1/1	1/0
1	0/0	2/2	2/0
2	3/3	2/2	2/0

$f^{(4)}$	0	1	2	3
0	1/1	1/1	2/2	2/0
1	4/4	5/5	2/2	2/0
2	4/4	3/3	3/3	3/0

$f^{(5)}$	0	1
0	0	0
1	0	1

$f^{(6)}$	0	1	2
0	1	1	2
1	4	5	2
2	4	3	3

$f^{(7)}$	4	2	0
3	4	3	3
1	4	2	1
0	4	2	0

$f^{(8)}$	0	1
0	0	1
1	0	2
2	3	2
3	3	1

Figure 1: Examples. Inputs for A are shown to the right, inputs for B on top. For asymmetric functions, outputs are denoted y_A/y_B; for symmetric functions only the common output of both parties is listed.

References

[ABDR04] Andris Ambainis, Harry Buhrman, Yevgeniy Dodis, and Hein Röhrig. Multiparty quantum coin flipping. In *IEEE Conference on Computational Complexity*, pages 250–259. IEEE Computer Society, 2004.

[BCMS99] Gilles Brassard, Claude Crépeau, Dominic Mayers, and Louis Salvail. Defeating classical bit commitments with a quantum computer. Los Alamos preprint archive quant-ph/9806031, May 1999.

[BMM99] Amos Beimel, Tal Malkin, and Silvio Micali. The all-or-nothing nature of two-party secure computation. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO'99*, volume 1666 of *LNCS*, pages 80–97. Springer, 1999.

[BT07] Anne Broadbent and Alain Tapp. Information-theoretic security without an honest majority. In *ASIACRYPT*, pages 410–426, 2007.

[CCM02] Christian Cachin, Claude Crépeau, and Julien Marcil. Oblivious transfer with a memory-bounded receiver. In *34th Annual ACM Symposium on Theory of Computing, Proceedings of STOC 2002*, pages 493–502. ACM Press, 2002.

[Cle86] Richard Cleve. Limits on the security of coin flips when half the processors are faulty. In *STOC '86: Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 364–369, New York, NY, USA, 1986. ACM Press.

[Cle90] Richard Cleve. Controlled gradual disclosure schemes for random bits and their applications. In Gilles Brassard, editor, *Advances in Cryptology, Proceedings of CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 573–588. Springer-Verlag, 1990.

[CM97] Christian Cachin and Ueli Maurer. Unconditional security against memory-bounded adversaries. In Burton S. Kaliski Jr., editor, *Advances in Cryptology—CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 292–306. Springer-Verlag, 17–21 August 1997.

[DFSS05] Ivan Damgård, Serge Fehr, Louis Salvail, and Christian Schaffner. Cryptography in the bounded quantum-storage model. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 449–458. IEEE Computer Society, 2005.

[DM04] Stefan Dziembowski and Ueli M. Maurer. On generating the initial key in the bounded-storage model. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 126–137. Springer, 2004.

[Gol04] Oded Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, 2004.

[HR07] Iftach Haitner and Omer Reingold. Statistically-hiding commitment from any one-way function. In David S. Johnson and Uriel Feige, editors, *STOC*, pages 1–10. ACM, 2007.

[IKLP06] Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. On combining privacy with guaranteed output delivery in secure multiparty computation. In C. Dwork, editor, *Advances in Cryptology — CRYPTO '06*, volume 4117 of *Lecture Notes in Computer Science*, pages 483–500. Springer, 2006.

[Kat06] Jonathan Katz. On achieving the "best of both worlds" in secure multiparty computation. Cryptology ePrint Archive, Report 2006/455, 2006. `http://eprint.iacr.org/`.

[Kil91] Joe Kilian. A general completeness theorem for two-party games. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, STOC'91*, pages 553–560, New York, 1991. ACM Press.

[Kil00] Joe Kilian. More general completeness theorems for secure two-party computation. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, STOC'00*, pages 316–324, New York, 2000. ACM Press.

[KMQ08] Daniel Kraschewski and Jörn Müller-Quade. Completeness theorems with constructive proofs for symmetric, asymmetric and general 2-party-functions. Unpublished Manuscript, 2008. Online available at `http://iks.ira.uka.de/eiss/completeness`.

[Kus92] Eyal Kushilevitz. Privacy and communication complexity. *SIAM Journal on Discrete Mathematics*, 5(2):273–284, 1992.

[LC97] H.-K. Lo and H. F. Chau. Is quantum bit commitment really possible? *Physical Review Letters*, 78:3410 – 3413, 1997.

[May97] D. Mayers. Unconditionally secure bit commitment is impossible. *Phys. Rev. Letters*, 78:3414–3417, 1997. A previous version was published at PhysComp96.

[MQ05] Jörn Müller-Quade. Temporary assumptions—quantum and classical. In *The 2005 IEEE Information Theory Workshop on Theory and Practice in Information-Theoretic Security*, pages 31–33, 2005.

[MQU07] J. Müller-Quade and D. Unruh. Long-term security and universal composability. In *Theory of Cryptography, Proceedings of TCC 2004*, Lecture Notes in Computer Science. Springer-Verlag, 2007. to appear.

[Rab03] Michael O. Rabin. Hyper-encryption by virtual satellite. Science Center Research Lecture Series, December 2003. Online available at `http://athome.harvard.edu/dh/hvs.html`.

A Security Models

We use a simulation based stand-alone model of security (as opposed to a universally composable model) with synchronous message passing where parties are connected via secure channels as for instance described in [Gol04]. Security is defined by comparing a real protocol with an ideal specification, where a trusted third party computes the function in question.[6] This comparison is done by means of a distinguisher D, that attempts to tell the ideal and the real setting apart. In the following we give a more formal definition of security.

Let $\mathfrak{P} = \{P_1, \ldots, P_n\}$ be a set of n parties and let $\mathfrak{F}_n$ designate the set of all n-party functions

$$f : \begin{cases} \mathcal{X}_1 \times \ldots \times \mathcal{X}_n \to \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_n \\ (x_1, \ldots, x_n) \mapsto f(x_1, \ldots, x_n) = (y_1, \ldots, y_n). \end{cases} \tag{5}$$

where the input and output sets $\mathcal{X}_1, \ldots, \mathcal{X}_n, \mathcal{Y}_1, \ldots, \mathcal{Y}_n \subset \mathbb{N}$ for each of the n parties are finite. Furthermore let $\mathfrak{F} := \bigcup_{n \geq 1} \mathfrak{F}_n$ denote the set of all multi-party functions. Now assume the parties in $\mathfrak{P}$ wish to compute a concrete function $f \in \mathfrak{F}_n$.[7] Throughout this work, we will mostly consider the case $n = 2$ and $\mathfrak{P} = \{P_A, P_B\}$. For convenience we often refer to parties by their index rather then by their full designation, writing for instance A instead of P_A.

We may now fix a set of corrupted parties $\mathfrak{C} \subset \mathfrak{P}$ and an adversary E that controls them. The set honest parties $\mathfrak{H}$ not controlled by the adversary is then $\mathfrak{H} := \mathfrak{P} \setminus \mathfrak{C}$. Furthermore we define a distinguisher D that determines the inputs x_i for all parties and after termination of the protocol receives their respective outputs y_i $(i \in \mathfrak{H} \cup \{\mathsf{E}\})$ where we designate input and output of the adversary by x_E and y_E respectively.

In the real case the uncorrupted parties $\mathfrak{H} := \mathfrak{P} \setminus \mathfrak{C}$ now run protocol π on the given inputs with the adversary E using (communication) resources R (in this work generally only a secure channel). The outputs of all parties are passed to D. We write $\mathsf{D}(\mathsf{E} \circ \pi_\mathfrak{H} \circ R)$ for this setup.

In the ideal case the specification and inputs of the adversary E are passed to a simulator S, the remaining inputs are passed to the ideal functionality I (which may depend on the set of corrupted parties $\mathfrak{C}$). The outputs of the uncorrupted parties $\mathfrak{H}$ are given directly to D, the outputs of the corrupted parties $\mathfrak{C}$ are passed to S, which then produces outputs for D. We write $\mathsf{D}(\mathsf{S}(\mathsf{E}) \circ I)$ for this setup.

Finally the distinguisher D has to output a bit $d \in \{0, 1\}$, which can be regarded as the distinguisher's guess if it is connected to the real system $\mathsf{E} \circ \pi \circ R$ or the ideal system $\mathsf{S}(\mathsf{E}) \circ I$. The protocol π is then regarded as secure with respect to the ideal specification I, if the distinguisher D has essentially no advantage in distinguishing the two settings:

Definition A.1 ((Stand-Alone) Security)**.** We call a protocol $\pi \circ R$ running on resources R stand-alone secure with respect to the ideal system I under a given advantage function ϵ and classes $\mathcal{D}$ of distinguishers, $\mathcal{E}$ of adversaries and $\mathcal{S}$ of simulators if and only if

$$\begin{aligned} &\forall \mathfrak{C} \subset \mathfrak{P}\, \forall \mathsf{E} \in \mathcal{E}\, \exists \mathsf{S} \in \mathcal{S}\, \forall \mathsf{D} \in \mathcal{D} \ : \\ &\qquad \Delta_\mathsf{D}(\mathsf{S}(\mathsf{E}) \circ I, \mathsf{E} \circ \pi \circ R) \\ &\qquad = \Delta(\mathsf{D}(\mathsf{S}(\mathsf{E}) \circ I), \mathsf{D}(\mathsf{E} \circ \pi \circ R)) \\ &\qquad = |P(\mathsf{D}(\mathsf{S}(\mathsf{E}) \circ I) = 1) - P(\mathsf{D}(\mathsf{E} \circ \pi \circ R) = 1)| < \epsilon(\kappa) \end{aligned}$$

where Δ denotes statistical distance. We then say the protocol π running on resources R is as secure as the ideal system I, written $\pi \circ R \succcurlyeq I$. We generally omit the resources R if they are clear from the context. $\Diamond$

[6] For a treatment of long-term security in the universal composability setting see [MQU07].

[7] In this work we take the function f to be independent of the security parameter κ. This is the natural and most relevant case for applications. Our proofs would however still hold for a family of functions f_κ, where the input domain grows at most polynomially fast in the security parameter κ.

Different security paradigms are now formalized by choosing specific sets $\mathcal{D}$, $\mathcal{E}$, $\mathcal{S}$ of distinguishers, adversaries and simulators and an advantage function $\epsilon(\kappa)$ or by specifying a specific ideal system I_f computing a function f.

We consider both computational (CO) security, where distinguishers, adversaries and simulators must be efficient[8] ($\mathcal{D}, \mathcal{E}, \mathcal{S} \subseteq \mathsf{Poly}$) and information-theoretic (IT) security where distinguishers, adversaries and simulators are arbitrary unbounded algorithms ($\mathcal{D}, \mathcal{E}, \mathcal{S} \subseteq \mathsf{Algo}$). Furthermore, we investigate long-term (LT) security where adversaries and simulators must be efficient ($\mathcal{E}, \mathcal{S} \subseteq \mathsf{Poly}$) but distinguishers may be unbounded ($\mathcal{D} \subseteq \mathsf{Algo}$), formalizing that we expect computational assumptions to hold only for the duration of the protocol. In all three cases the advantage $\epsilon(\kappa)$ is chosen to be negligible in the security parameter κ. Here $\epsilon(\kappa)$ being negligible means

$$\forall p \in \mathbb{Z}[X]\ \exists \kappa_0 \in \mathbb{N}\ \forall \kappa \in \mathbb{N}\ :\ \kappa > \kappa_0 \implies \epsilon(\kappa) < \frac{1}{p(\kappa)},$$

and is denoted $\epsilon < \mathsf{negl}$ and we write $\epsilon_1 \approx \epsilon_2$ iff $|\epsilon_1 \approx \epsilon_2| < \mathsf{negl}$. If in the IT case we fix an advantage of $\epsilon = 0$ we arrive at perfect (PF) security. We may vary the IT and PF cases by demanding efficient simulators (ITE, PFE).

In summary we consider the following basic security paradigms:

Perfect (PF) security: Distinguishers, adversaries and simulators are arbitrary algorithms $\mathcal{D}^{\mathsf{PF}} = \mathcal{E}^{\mathsf{PF}} = \mathcal{S}^{\mathsf{PF}} = \mathsf{Algo}$, the advantage $\epsilon(\kappa) = 0$ is zero. Notation: $\pi \succcurlyeq^{\mathsf{PF}} I$.

Information-theoretic (IT) security: Distinguishers, adversaries and simulators are arbitrary algorithms $\mathcal{D}^{\mathsf{IT}} = \mathcal{E}^{\mathsf{IT}} = \mathcal{S}^{\mathsf{IT}} = \mathsf{Algo}$, the advantage $\epsilon(\kappa) < \mathsf{negl}$ is negligible. Notation: $\pi \succcurlyeq^{\mathsf{IT}} I$.

PF security with efficient simulator (PFE): Distinguishers and adversaries are arbitrary algorithms $\mathcal{D}^{\mathsf{PFE}} = \mathcal{E}^{\mathsf{PFE}} = \mathsf{Algo}$, simulators are efficient $\mathcal{S}^{\mathsf{PFE}} = \mathsf{Poly}$, the advantage $\epsilon(\kappa) = 0$ is zero. Notation: $\pi \succcurlyeq^{\mathsf{PFE}} I$.

IT security with efficient simulator (ITE): Distinguishers and adversaries are arbitrary algorithms $\mathcal{D}^{\mathsf{ITE}} = \mathcal{E}^{\mathsf{ITE}} = \mathsf{Algo}$, simulators are efficient $\mathcal{S}^{\mathsf{ITE}} = \mathsf{Poly}$, the advantage $\epsilon(\kappa) < \mathsf{negl}$ is negligible. Notation: $\pi \succcurlyeq^{\mathsf{ITE}} I$.

Computational (CO) security: Distinguishers, adversaries and simulators are efficient $\mathcal{D}^{\mathsf{CO}} = \mathcal{E}^{\mathsf{CO}} = \mathcal{S}^{\mathsf{CO}} = \mathsf{Poly}$, the advantage $\epsilon(\kappa) < \mathsf{negl}$ is negligible. Notation: $\pi \succcurlyeq^{\mathsf{CO}} I$.

Long-term (LT) security: Distinguishers are arbitrary algorithms $\mathcal{D}^{\mathsf{LT}} = \mathsf{Algo}$, adversaries and simulators are efficient $\mathcal{E}^{\mathsf{LT}} = \mathcal{S}^{\mathsf{LT}} = \mathsf{Poly}$, the advantage $\epsilon(\kappa) < \mathsf{negl}$ is negligible. Notation: $\pi \succcurlyeq^{\mathsf{LT}} I$.

Note that a first type of security given through $\mathcal{D}_1$, $\mathcal{E}_1$, $\mathcal{S}_1$ implies a second type of security given through $\mathcal{D}_2$, $\mathcal{E}_2$, $\mathcal{S}_2$ if $\mathcal{D}_1 \supseteq \mathcal{D}_2$, $\mathcal{E}_1 \supseteq \mathcal{E}_2$, $\mathcal{S}_1 \subseteq \mathcal{S}_2$. We obtain the following implications

$$\begin{tikzcd} \mathsf{PFE} \arrow[r] \arrow[d] & \mathsf{ITE} \arrow[r] \arrow[d] & \mathsf{CO} \\ \mathsf{PF} \arrow[r] & \mathsf{IT} & \end{tikzcd} \tag{6}$$

We refine these security paradigms further by defining adversarial models, i.e. restrictions that we can impose on the adversaries and simulators for any of the above paradigms:

Active (ACT) adversaries: Adversaries and simulators not restricted further, $\mathcal{E}_{\mathsf{act}} = \mathcal{S}_{\mathsf{act}} = \mathsf{Algo}$.

Semi-honest (SH) adversaries: Adversaries in the class $\mathcal{E}_{\mathsf{sh}}$ are restricted to generate messages according to the prescribed protocol π with the inputs provided by the distinguisher D, simulators are not restricted further $\mathcal{S}_{\mathsf{act}} = \mathsf{Algo}$.

[8] By efficient we mean polynomially bounded in the security parameter κ.

Passive (PA) adversaries: Adversaries in the class $\mathcal{E}_{\mathsf{pas}} = \mathcal{E}_{\mathsf{sh}}$ are restricted to generate messages according to the prescribed protocol π with the inputs provided by the distinguisher D, simulators in the class $\mathcal{S}_{\mathsf{pas}}$ are restricted to forward the inputs provided by the distinguisher D to the ideal functionality I.

Security paradigms and adverserial models as defined above are combined by intersecting their defining sets, i.e. semi-honest IT security is described by $\mathcal{D}^{\mathsf{IT}}_{\mathsf{sh}} = \mathcal{D}^{\mathsf{IT}}$, $\mathcal{S}^{\mathsf{IT}}_{\mathsf{sh}} = \mathcal{S}^{\mathsf{IT}} \cap \mathcal{S}_{\mathsf{sh}}$, $\mathcal{E}^{\mathsf{IT}}_{\mathsf{sh}} = \mathcal{E}^{\mathsf{IT}} \cap \mathcal{E}_{\mathsf{sh}}$, $\epsilon(\kappa) < \mathsf{negl}$ and denoted $\pi \succcurlyeq^{\mathsf{IT}}_{\mathsf{sh}} I$.

Among the adversarial models we find the following implications

$$\mathsf{act} \longrightarrow \mathsf{sh} \longleftarrow \mathsf{pas} \tag{7}$$

We can now formalize security requirements under each of the definitions above by providing an appropriate ideal system. In the following we describe ideal systems that formalize

1. robustly computing a function f
2. non-robustly computing a function f (security with abort)
3. non-robustly computing a function f where only a specific party can abort (security with designated aborter)

Let a function

$$f : \begin{cases} \mathcal{X}_1 \times \ldots \times \mathcal{X}_n \to \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_n \\ (x_1, \ldots, x_n) \mapsto f(x_1, \ldots, x_n) = (y_1, \ldots, y_n) \end{cases} \tag{8}$$

with n inputs and n outputs, corresponding to the n parties in $\mathfrak{P}$, be given and $\mathfrak{C} \subset \mathfrak{P}$ be a set of corrupted players.

Definition A.2 (Security (with Robustness))**.** We define an ideal functionality I_f that formalizes computing f securely (including robustness) as follows:

First $I_f(\mathfrak{C})$ takes inputs $x_i \in \mathcal{X}_i$ from each party P_i. If no valid input is received from P_i then a default value x_i^{def} for x_i is assumed. Now $I_f(\mathfrak{C})$ computes $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$ and outputs y_i to P_i for all $P_i \in \mathfrak{P}$. $\Diamond$

Unfortunately, in protocols tolerating a majority of dishonest parties robustness and even quit-fairness cannot generally be guaranteed [Cle86, Cle90, IKLP06, Kat06]. In other words, an adversary can possibly abort the protocol even after obtaining (part of) the protocol result, while preventing the honest parties from learning their result. Although such a scenario is clearly less desirable then fair or even robust computation, we are still interested in security in presence dishonest majorities. Thus, we devise an ideal system that explicitly allows for this type of behavior:

Definition A.3 (Security with Abort)**.** We define an ideal functionality $I_f^{\mathsf{ab}}(\mathfrak{C})$ that formalizes computing a function f securely with abort as follows:

First $I_f^{\mathsf{ab}}(\mathfrak{C})$ takes inputs $x_i \in \mathcal{X}_i$ from each party P_i. If no valid input x_i is received from P_i then a default value x_i^{def} for x_i is assumed. Now $I_f^{\mathsf{ab}}(\mathfrak{C})$ computes $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$ and outputs y_i for $P_i \in \mathfrak{C}$ to the ideal adversary (simulator) S. The ideal functionality $I_f^{\mathsf{ab}}(\mathfrak{C})$ then awaits an input $o \in \{0, 1\}$ from the the simulator S. On $o = 1$ the functionality $I_f^{\mathsf{ab}}(\mathfrak{C})$ distributes y_i to all parties P_i, on $o = 0$ it sends $\perp$ to all parties P_i and terminates. $\Diamond$

As mentioned above, in protocols tolerating up to $n - 1$ out of n corrupted parties successful termination cannot generally be guaranteed. A single corrupted party can possibly abort the protocol. However, it is also known that CO secure SFE protocols can be designed in such a way that only a single party, to which we refer as the designated aborter, can abort the protocol *and* (computationally) learn anything about the result of the computation. In other words, only the designated aborter can abort the protocol conditional on his output. Aborts from other players are no different from refusal to participate in the protocol. This notion of *computational security with designated aborter* is of practical interest, because we may in an application setting have a party which is not fully trusted but can be relied upon not to abort the protocol. We hence formalize this notion in the following definition:

Definition A.4 (Security with Designated Aborter (DA))**.** We define an ideal functionality $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ that formalizes securely computing a function f with designated aborter P_j as follows:

First $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ takes inputs $x_i \in \mathcal{X}_i$ from each party P_i. If no valid input is received from P_i then a default value x_i^{def} for x_i is assumed. Now $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ computes $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$. If $P_j \in \mathfrak{C}$ the functionality $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ outputs y_i for $P_i \in \mathfrak{C}$ to the ideal adversary (simulator) S. Else it only outputs y_j to P_j. $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ then awaits an input $o \in \{0, 1\}$ from the designated aborter P_j. On $o = 1$ the ideal system $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ distributes y_i to all parties P_i. On $o = 0$ $I_f^{\mathsf{des}(j)}(\mathfrak{C})$ sends $\perp$ to all parties and terminates.

We generally set $j = 1$ and drop it from the notation. $\diamondsuit$

Naturally we investigate how the notions of LT security and computational security with designated aborter combine. To this end we define

Definition A.5 (Long Term Security with Designated Aborter (LTS-DA))**.** A protocol π computes a function f with long-term security with designated aborter (LTS-DA) if it computes f with CO security with designated aborter and if the protocol π' where the reply of an honest P_1 after receiving its output is fixed to ok computes f with LT security with abort. $\diamondsuit$

Note that in LTS-DA a long-term protection of secrets is only guaranteed for protocols which terminate. In case of an abort an adversary may in the long term learn intermediate results (by breaking the computational assumption).

B Proof of Lemma 3.4

We have to show: $f \in \mathfrak{T}_{\mathsf{2pas}} \iff f \in \mathfrak{T}'_{\mathsf{2pas}}$. We address each implication separately.

$f \in \mathfrak{T}'_{\mathsf{2pas}} \implies f \in \mathfrak{T}_{\mathsf{2pas}}$**:** We proceed by induction over the size of the input space $|\mathcal{X}_A \times \mathcal{X}_B|$. Now $f \in \mathfrak{T}'_{\mathsf{2pas}}$ implies that f is either $f \in \mathfrak{T}_{\mathsf{2loc}}$ locally computable, in which case trivially $f \in \mathfrak{T}_{\mathsf{2pas}}$ passively trivial and we are done, or f has a K-cut.

In the second case, by definition of the class $\mathfrak{T}'_{\mathsf{2pas}}$ and wlog (else interchange A and B) the set $\mathcal{X}_B$ has a partition $\mathcal{X}_B^{(1)}, \mathcal{X}_B^{(2)}$ such that

$$\forall x_A \in \mathcal{X}_A \ : \ f_A(x_A, \mathcal{X}_B^{(1)}) \cap f_A(x_A, \mathcal{X}_B^{(2)}) = \emptyset$$

and the functions $f^{(1)} := f|_{\mathcal{X}_A \times \mathcal{X}_B^{(1)}}$, $f^{(2)} := f|_{\mathcal{X}_A \times \mathcal{X}_B^{(2)}}$ are $f^{(1)}, f^{(2)} \in \mathfrak{T}'_{\mathsf{2pas}}$. By induction hypothesis then $f^{(1)}, f^{(2)} \in \mathfrak{T}_{\mathsf{2pas}}$ as $|\mathcal{X}_A \times \mathcal{X}_B^{(i)}| < |\mathcal{X}_A \times \mathcal{X}_B|$ ($i \in \{1, 2\}$). As such, there are protocols $\pi^{(1)}, \pi^{(2)}$ passively PF securely implementing $I_{f^{(1)}}, I_{f^{(2)}}$ under simulators $\mathsf{S}^{(1)}, \mathsf{S}^{(2)}$.

We now construct a protocol π by defining a first round where B sends A a message $m_1 \in \{1, 2\}$ chosen such that indicating wether B's input b is $b \in \mathcal{X}_B^{(1)}$ or $b \in \mathcal{X}_B^{(2)}$. Subsequently both parties proceed to run $\pi^{(m_1)}$.

It remains to prove this protocol π secure by providing a appropriate simulator S both for the case where A is corrupted and for the case where B is corrupted.

We begin with corrupted A. The simulator S_A fixes the randomness of the adversary E (which it uses as a black-box) and feeds it the input x_E provided by the distinguisher D. Then the simulator S_A extracts the input a from x_E and inputs it to the ideal system I_f as prescribed for the passive setting, in return receiving output $y_A = f_A(a,b)$. Now as $f \in \mathfrak{T}'_{\mathsf{2pas}}$ we have $f_A(a, \mathcal{X}_B^{(1)}) \cap f_A(a, \mathcal{X}_B^{(2)}) = \emptyset$. So the simulator S_A can uniquely determine $m_1 \in \{1,2\}$ such that $y_A \in f_A(a, \mathcal{X}_B^{(m_1)})$. Finally the simulator S_A inputs m_1 to the adversary E and runs $\mathsf{S}_A^{(m_1)}(\mathsf{E})$, forwarding all inputs and outputs. The correctness of the simulation stems from the fact that the simulator S_A can correctly discern the first message m_1 from the output y_A of the ideal functionality I_f. After simulating this message, the remainder of the simulation can be delegated to the subsimulator $\mathsf{S}_A^{(m_1)}$ that is guaranteed by our inductive argument.

We now turn to corrupted B. The simulator S_B fixes the randomness of the adversary E feeds it the input x_E provided by the distinguisher D and runs the adversary E until E outputs the message m_1. Now the simulator S_B runs $\mathsf{S}_B^{(m_1)}(\mathsf{E})$ simply forwarding all inputs and outputs. This simulates the real execution, as depending on message m_1 the protocol $\pi^{(m_1)}$ is executed by the honest party A. The interaction with this protocol now is faithfully simulated by $\mathsf{S}_B^{(m_1)}$.

Note that we actually prove passively PF security above. The computation of locally computable functions is perfectly secure (induction base) and in the above argument, if the subsimulators are perfectly secure, so are the resulting simulators (induction step).

$f \in \mathfrak{T}_{\mathsf{2pas}} \implies f \in \mathfrak{T}'_{\mathsf{2pas}}$ Towards a contradition assume a counterexample $f \in \mathfrak{T}_{\mathsf{2pas}}$ minimal in the size of the input space $|\mathcal{X}_A \times \mathcal{X}_B|$ such that $f \notin \mathfrak{T}'_{\mathsf{2pas}}$. Now consider an arbitrary restriction $\widetilde{f} := f|_{\widetilde{\mathcal{X}}_A \times \widetilde{\mathcal{X}}_B}$ of f to subsets $\widetilde{\mathcal{X}}_A \subset \mathcal{X}_A$, $\widetilde{\mathcal{X}}_B \subset \mathcal{X}_B$ where at least one inclusion is strict. As $f \in \mathfrak{T}_{\mathsf{2pas}}$, so is $\widetilde{f} \in \mathfrak{T}_{\mathsf{2pas}}$. Indeed this can be seen by using exactly the same protocol and simulator as for f, and merely restricting the input domain. By the minimality of f we furthermore have $\widetilde{f} \in \mathfrak{T}'_{\mathsf{2pas}}$. Hence, by the assumption $f \notin \mathfrak{T}'_{\mathsf{2pas}}$ the function f cannot be locally computable and has no K-cut:

1. $\exists x'_A, x''_A \in \mathcal{X}_A, x'_B, x''_B \in \mathcal{X}_B \ : \ f_A(x'_A, x'_B) \neq f_A(x'_A, x''_B) \vee f_B(x'_A, x'_B) \neq f_B(x''_A, x'_B)$ and
2. for any partition $\mathcal{X}'_A \cup \mathcal{X}''_A = \mathcal{X}_A$ there is an $x_B \in \mathcal{X}_B$ such that $f_B(\mathcal{X}'_A, x_B) \cap f_B(\mathcal{X}''_A, x_B) \neq \emptyset$ and
3. for any partition $\mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B$ there is an $x_A \in \mathcal{X}_A$ such that $f_A(x_A, \mathcal{X}'_B) \cap f_A(x_A, \mathcal{X}''_B) \neq \emptyset$.

Now as $f \in \mathfrak{T}_{\mathsf{2pas}}$ we have an efficient passively secure protocol π computing f. Then the number of rounds of protocol π is bounded by a polynomial $p_\pi(\kappa)$ in the security parameter κ. So we can wlog "pad" all protocol executions with dummy messages to have the same length $p_\pi(\kappa)$. Denote the inputs of A and B by x_A and x_B and the random coins by c_A and c_B respectively. We then define the set of transcripts

$$\Pi := \pi(\mathcal{X}_A, \mathcal{X}_B) = \{t = m_1, m_2, \ldots, m_{p_\pi(\kappa)} \mid \exists\, x_A, c_A, x_B, c_B \ : \ t = \pi(x_A, c_A, x_B, c_B)\} \tag{9}$$

$$\begin{aligned}\Pi_r := &\pi(\mathcal{X}_A, \mathcal{X}_B)|_r = \{t = m_1, m_2, \ldots, m_r \mid \exists\, m_{r+1}, \ldots, m_{p_\pi(\kappa)} \ : \ t, m_{r+1}, \ldots, m_{p_\pi(\kappa)} \in \Pi\} \\ =&\{t = m_1, m_2, \ldots, m_r \mid \exists\, x_A, c_A, x_B, c_B \ : \ t = \pi(x_A, c_A, x_B, c_B)|_r\}\end{aligned} \tag{10}$$

We define random variables X_A, X_B, Y_A, Y_B for the inputs and outputs of A and B respectively. Then we let $T \in \Pi$ denote the random variable for the transcript, and $T_r \in \Pi_r$ the random variable on transcript prefixes of length r. We name the inputs $\mathcal{X}_A = \{a_1, \ldots, a_{|\mathcal{X}_A|}\}$ for A and $\mathcal{X}_B = \{b_1, \ldots, b_{|\mathcal{X}_B|}\}$ for B.

We show that for any $a_i \in \mathcal{X}_A$ and $b_j \in \mathcal{X}_B$ the statistical distance

$$\Delta(\Pr_{T|X_A,X_B}(\cdot, a_1, b_1), \Pr_{T|X_A,X_B}(\cdot, a_i, b_j)) < \mathsf{negl} \tag{11}$$

of the distributions $\Pr_{T|X_A,X_B}(\cdot, a_1, b_1)$ and $\Pr_{T|X_A,X_B}(\cdot, a_i, b_j)$ on the transcripts $t \in \Pi$ is negligible in the security parameter κ. We proceed by induction over the number of protocol rounds r and show that for any r

$$\delta_r := \max_{i,j} \Delta(\Pr_{T_r|X_A,X_B}(\cdot, a_1, b_1), \Pr_{T_r|X_A,X_B}(\cdot, a_i, b_j)) < \mathsf{negl}. \tag{12}$$

As a base case serves $r = 0$ where $\Pi_0 = \{\varepsilon\}$ and thus

$$\delta_0 = \max_{i,j} \Delta(\Pr_{T_0|X_A,X_B}(\cdot, a_1, b_1), \Pr_{T_0|X_A,X_B}(\cdot, a_i, b_j)) = 0. \tag{13}$$

As induction hypothesis we use $\delta_{r-1} < \mathsf{negl}$.

Now let $t_r = (m_1, \ldots, m_r) \in \Pi_r$ and define $t_{r-1} = (m_1, \ldots, m_{r-1}) \in \Pi_{r-1}$. Wlog the message in round r travels from B to A (the other case works analogously). Furthermore, as $f \notin \mathfrak{T}'_{\mathsf{2pas}}$ and minimal, f has no K-cut as discussed above and (by point 3 above) we can impose an ordering on $\mathcal{X}_B$ such that for any $1 < j \leq |\mathcal{X}_B|$ there is an $1 \leq k < j$ and an $a_{kj} \in \mathcal{X}_A$ such that $f_A(a_{kj}, b_k) = f_A(a_{kj}, b_j)$. By the security of π we must have for any r that

$$\Delta(\Pr_{T_r|X_A,X_B}(\cdot, a_{kj}, b_k), \Pr_{T_r|X_A,X_B}(\cdot, a_{kj}, b_j)) < \mathsf{negl} \tag{14}$$

This is because a simulator S_A for corrupted A sees exactly the same for inputs a_{kj}, b_k and a_{kj}, b_j and is thus unable to emulate two *non-negligibly different* distributions of transcripts. Now, since the message m_r travels from B to A and we have

$$\begin{aligned}
&\Delta(\Pr_{T_r|X_A,X_B}(\cdot, a_1, b_k), \Pr_{T_r|X_A,X_B}(\cdot, a_{kj}, b_k)) \\
&\quad = \sum_{t_r \in \Pi_r} |\Pr_{T_r|X_A,X_B}(t_r, a_1, b_k) - \Pr_{T_r|X_A,X_B}(t_r, a_{kj}, b_k)| \\
&\quad = \sum_{t_r \in \Pi_r} |\Pr_{M_r|T_{r-1},X_B}(m_r, t_{r-1}, b_k)\Pr_{T_{r-1}|X_A,X_B}(t_{r-1}, a_1, b_k) \\
&\qquad\quad - \Pr_{M_r|T_{r-1},X_B}(m_r, t_{r-1}, b_k)\Pr_{T_{r-1}|X_A,X_B}(t_{r-1}, a_{kj}, b_k)| \\
&\quad = \sum_{t_{r-1} \in \Pi_{r-1}} \underbrace{\sum_{m_r} \Pr_{M_r|T_{r-1},X_B}(m_r, t_{r-1}, b_k)}_{=1} \\
&\qquad\quad |\Pr_{T_{r-1}|X_A,X_B}(t_{r-1}, a_1, b_k) - \Pr_{T_{r-1}|X_A,X_B}(t_{r-1}, a_{kj}, b_k)| \\
&\quad = \Delta(\Pr_{T_{r-1}|X_A,X_B}(\cdot, a_1, b_k), \Pr_{T_{r-1}|X_A,X_B}(\cdot, a_{kj}, b_k)) \\
&\quad \leq \Delta(\Pr_{T_{r-1}|X_A,X_B}(\cdot, a_1, b_k), \Pr_{T_{r-1}|X_A,X_B}(\cdot, a_1, b_1)) \\
&\qquad\quad + \Delta(\Pr_{T_{r-1}|X_A,X_B}(\cdot, a_1, b_1), \Pr_{T_{r-1}|X_A,X_B}(\cdot, a_{kj}, b_k)) \\
&\quad \leq 2\delta_{r-1} < \mathsf{negl},
\end{aligned} \tag{15}$$

and analogously

$$\Delta(\Pr_{T_r|X_A,X_B}(\cdot, a_{kj}, b_j), \Pr_{T_r|X_A,X_B}(\cdot, a_i, b_j)) \tag{16}$$

$$= \sum_{t_r \in \Pi_r} |\Pr_{T_r|X_A,X_B}(t_r, a_{kj}, b_j) - \Pr_{T_r|X_A,X_B}(t_r, a_i, b_j)| \tag{17}$$

$$\leq 2\delta_{r-1} < \mathsf{negl}. \tag{18}$$

so we find

$$
\begin{aligned}
&\Delta(\Pr_{T_r|X_A,X_B}(\cdot,a_1,b_k),\Pr_{T_r|X_A,X_B}(\cdot,a_i,b_j)) && (19)\\
&\quad= \sum_{t_r\in\Pi_r} |\Pr_{T_r|X_A,X_B}(t_r,a_1,b_k) - \Pr_{T_r|X_A,X_B}(t_r,a_i,b_j)| && (20)\\
&\quad= \sum_{t_r\in\Pi_r} |\Pr_{T_r|X_A,X_B}(t_r,a_1,b_k) - \Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_k) && (21)\\
&\qquad\quad + \Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_k) - \Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_j) && (22)\\
&\qquad\quad + \Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_j) - \Pr_{T_r|X_A,X_B}(t_r,a_i,b_j)| && (23)\\
&\quad\le \sum_{t_r\in\Pi_r} |\Pr_{T_r|X_A,X_B}(t_r,a_1,b_k) - \Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_k)| && (24)\\
&\qquad\quad + \sum_{t_r\in\Pi_r} |\Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_k) - \Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_j)| && (25)\\
&\qquad\quad + \sum_{t_r\in\Pi_r} |\Pr_{T_r|X_A,X_B}(t_r,a_{kj},b_j) - \Pr_{T_r|X_A,X_B}(t_r,a_i,b_j)| && (26)\\
&\quad= \Delta(\Pr_{T_r|X_A,X_B}(\cdot,a_1,b_k),\Pr_{T_r|X_A,X_B}(\cdot,a_{kj},b_k)) && (27)\\
&\qquad\quad + \Delta(\Pr_{T_r|X_A,X_B}(\cdot,a_{kj},b_k),\Pr_{T_r|X_A,X_B}(\cdot,a_{kj},b_j)) && (28)\\
&\qquad\quad + \Delta(\Pr_{T_r|X_A,X_B}(\cdot,a_{kj},b_j),\Pr_{T_r|X_A,X_B}(\cdot,a_i,b_j)) && (29)\\
&\quad\overset{(14),(15),(16)}{\le} \mathsf{negl}+\mathsf{negl}+\mathsf{negl} && (30)
\end{aligned}
$$

Now by induction on j we finally obtain

$$\Delta(\Pr_{T_r|X_A,X_B}(\cdot,a_1,b_1),\Pr_{T_r|X_A,X_B}(\cdot,a_i,b_j)) < \mathsf{negl} \tag{31}$$

for any i, j and thus $\delta_r < \mathsf{negl}$. This concludes the induction step and thereby the proof of Eq. (11). Since we operate with negligible quantities here, it is important to note that the induction chains are all of length at most polynomial in the security parameter κ. But this is given, since we require the round number $p_\pi(\kappa)$ to be polynomially bounded and as for a given fixed function f the size of the input domains $|\mathcal{X}_A|$ and $|\mathcal{X}_B|$ are of course constant in κ.

As $f \notin \mathfrak{T}'_{\mathsf{2pas}}$ it is in particular not locally computable. Hence there is wlog (else interchange A and B) an $a \in \mathcal{X}_A$ and $b', b'' \in \mathcal{X}_B$ such that $f_A(a,b') \neq f_A(a,b'')$. From the above Eq. (11) it readily follows that

$$\Delta(\Pr_{T|X_A,X_B}(\cdot,a,b'),\Pr_{T|X_A,X_B}(\cdot,a,b'')) < \mathsf{negl}. \tag{32}$$

As the output of A is independent of B's input given the transcript, i.e. $\Pr_{Y_A|T,X_A,X_B} = \Pr_{Y_A|T,X_A}$ we find the following

$$\Delta(\Pr_{Y_A|X_A,X_B}(\cdot,a,b'),\Pr_{Y_A|X_A,X_B}(\cdot,a,b'')) \tag{33}$$

$$= \sum_{y_A\in\mathcal{Y}_A} |\Pr_{Y_A|X_A,X_B}(y_A,a,b') - \Pr_{Y_A|X_A,X_B}(y_A,a,b'')| \tag{34}$$

$$= \sum_{y_A\in\mathcal{Y}_A} |\sum_{t\in\Pi}\Pr_{Y_A,T|X_A,X_B}(y_A,t,a,b') - \sum_{t\in\Pi}\Pr_{Y_A,T|X_A,X_B}(y_A,t,a,b'')| \tag{35}$$

$$\leq \sum_{y_A\in\mathcal{Y}_A,t\in\Pi} |\Pr_{Y_A|T,X_A,X_B}(y_A,t,a,b')\Pr_{T|X_A,X_B}(t,a,b') - \Pr_{Y_A|T,X_A,X_B}(y_A,t,a,b'')\Pr_{T|X_A,X_B}(t,a,b'')| \tag{36}$$

$$= \sum_{y_A\in\mathcal{Y}_A,t\in\Pi} |\Pr_{Y_A|T,X_A}(y_A,t,a)\Pr_{T|X_A,X_B}(t,a,b') - \Pr_{Y_A|T,X_A}(y_A,t,a)\Pr_{T|X_A,X_B}(t,a,b'')| \tag{37}$$

$$= \sum_{y_A\in\mathcal{Y}_A,t\in\Pi} \Pr_{Y_A|T,X_A}(y_A,t,a)\ |\Pr_{T|X_A,X_B}(t,a,b') - \Pr_{T|X_A,X_B}(t,a,b'')| \tag{38}$$

$$= \sum_{t\in\Pi}\left(\sum_{y_A\in\mathcal{Y}_A}\Pr_{Y_A|T,X_A}(y_A,t,a)\right)|\Pr_{T|X_A,X_B}(t,a,b') - \Pr_{T|X_A,X_B}(t,a,b'')| \tag{39}$$

$$= \Delta(\Pr_{T|X_A,X_B}(\cdot,a,b'),\Pr_{T|X_A,X_B}(\cdot,a,b'')) < \mathsf{negl}. \tag{40}$$

This is in obvious contradiction to the correctness of the protocol π, so we must have $f \in \mathfrak{T}_{\mathsf{2pas}}$. Hence there is no minimal counterexample and the claim is proven.

C Dominance and Redundancy-Freeness

Definition C.1 (Dominance, Redundancy-Freeness [KMQ08])**.** Given a 2-party function $f : \mathcal{X}_A \times \mathcal{X}_B \to \mathcal{Y}_A \times \mathcal{Y}_B$ we say x_A *dominates* x'_A for $x_A, x'_A \in \mathcal{X}_A$, iff

1. for all $x_B \in \mathcal{X}_B$: $f_B(x_A, x_B) = f_B(x'_A, x_B)$ and
2. for all $x_B, x'_B \in \mathcal{X}_B$ $f_A(x'_A, x_B) \neq f_A(x'_A, x'_B) \implies f_A(x_A, x_B) \neq f_A(x_A, x'_B)$.

Analogously we define x_B dominates x'_B for $x_B, x'_B \in \mathcal{X}_B$.

We proceed to define sets of dominating inputs

$$\widetilde{\mathcal{X}_i} := \{\mathcal{X} \subseteq \mathcal{X}_i \mid \forall x' \in \mathcal{X}_i \exists x \in \mathcal{X} : x \text{ dominates } x'\}.$$

We define *dominating sets* $\hat{\mathcal{X}}_A$, $\hat{\mathcal{X}}_B$ as (some) elements of minimal cardinality in $\widetilde{\mathcal{X}_A}$, $\widetilde{\mathcal{X}_B}$. We then call $\hat{f} := f|_{\hat{\mathcal{X}}_A\times\hat{\mathcal{X}}_B}$ the *redundancy-free version* of f. Furthermore for $x_A \in \mathcal{X}_A$ and $x_B \in \mathcal{X}_B$ let $\hat{x}_A \in \hat{\mathcal{X}}_A$ and $\hat{x}_B \in \hat{\mathcal{X}}_B$ be the (unique) elements that dominate x_A respectively x_B. ◊

The redundancy-free version $\hat{f}$ of f is then uniquely defined up to a renaming of input and output values (also see Sec. 8). To see this, note that domination is a reflexive and transitive relation. Furthermore it is antisymmetric up to renaming of input and output symbols. Hence two different dominating sets $\hat{\mathcal{X}}_i$ and $\hat{\mathcal{X}}'_i$ are simply sets of maximal elements under the domination relation and equal up to renaming of input and output values.

D Proof of Lemma 4.2

We have to show that I_f and $I_{\hat{f}}$ are locally mutually reducible. We prove each direction separately.

Let I_f be given. We implement $I_{\hat{f}}$ using a protocol $\pi_{\hat{f}}$ that simply restricts the input space for I_f. Correctness (for the all-honest case) is obvious. To prove security we have to provide a simulator S. We only consider the case where A is corrupted, as the argument for corrupted B is fully analogous. The simulator S_A simply runs the adversary E until E produces an input a intended for I_f. S_A then inputs $\hat{a}$ to $I_{\hat{f}}$ and receives an output $\hat{y}_A = \hat{f}_A(\hat{a}, b)$ where b is the input of honest B. Now by definition of $\hat{a}$ we have

1. $f_B(\hat{a}, b) = f_B(a, b)$ and
2. for all $x_B, x'_B \in \mathcal{X}_B$ $f_A(a, x_B) \neq f_A(a, x'_B) \implies f_A(\hat{a}, x_B) \neq f_A(\hat{a}, x'_B)$.

By the second point S_A can deduce $y_A = f(a, b)$ from $\hat{y}_A$. The simulator S_A then feeds y_A to the adversary E and forwards its output y_E to the distinguisher D. This clearly results in a perfectly indistinguishable interaction for E and hence in indistinguishable output y_E. Due to the first point above the simulation is also indistinguishable by B's output y_B, thus $\pi_{\hat{f}} \circ I_f$ indeed perfectly securely implements $I_{\hat{f}}$ and the simulator S_A is efficient.

Now let $I_{\hat{f}}$ be given. We describe the protocol π for implementing I_f from $I_{\hat{f}}$ by describing A's protocol π_A. π_B works analogously. The protocol π_A takes input a and in turn inputs $\hat{a}$ to $I_{\hat{f}}$, receiving $\hat{y}_A = \hat{f}_A(\hat{a}, \hat{b})$ in return. Now by definition of $\hat{b}$ and $\hat{f}$ we have $\hat{y}_A = \hat{f}_A(\hat{a}, \hat{b}) = f_A(\hat{a}, b)$. By definition of $\hat{a}$ then for all $x_B, x'_B \in \mathcal{X}_B$ $f_A(a, x_B) \neq f_A(a, x'_B) \implies f_A(\hat{a}, x_B) \neq f_A(\hat{a}, x'_B)$. Once again this implies that π_A can recover $y_A = f(a, b)$ from $\hat{y}_A$. Finally the protocol π_A outputs y_A to the distinguisher. The correctness of the protocol for the all honest case is immediate. The security proof is also trivial: The simulator S_A simply restricts the input space for I_f to $\hat{\mathcal{X}}_A$ and analogously for B.

E Proof of Lemma 4.3

By Lem. 4.2 it is sufficient to show for redundancy-free functions f where $f = \hat{f}$ that $f \in \mathfrak{T}_{2\mathsf{sh}} \iff f \in \mathfrak{T}_{2\mathsf{pas}}$. For all other (not redundancy-free) functions we then find $f \in \mathfrak{T}_{2\mathsf{sh}} \overset{(4.2)}{\iff} \hat{f} \in \mathfrak{T}_{2\mathsf{sh}} \iff \hat{f} \in \mathfrak{T}_{2\mathsf{pas}}$ as claimed. So in the following let $f = \hat{f}$ be redundancy-free. We show $f \in \mathfrak{T}_{2\mathsf{sh}} \iff f \in \mathfrak{T}_{2\mathsf{pas}}$ by proving the two implications separately.

$f \in \mathfrak{T}_{2\mathsf{pas}} \implies f \in \mathfrak{T}_{2\mathsf{sh}}$: Since passive security implies semi-honest security as shown in Section 2 this is clear.

$f \in \mathfrak{T}_{2\mathsf{sh}} \implies f \in \mathfrak{T}_{2\mathsf{pas}}$: Towards a contradiction let f be redundancy free and $f \in \mathfrak{T}_{2\mathsf{sh}}$, but $f \notin \mathfrak{T}_{2\mathsf{pas}}$. Now consider a corrupted A and an adversary $\mathsf{E} \in \mathcal{E}^{\mathsf{IT}}_{\mathsf{sh}} = \mathcal{E}^{\mathsf{IT}}_{\mathsf{pas}}$. By definition of $\mathcal{E}^{\mathsf{IT}}_{\mathsf{pas}}$ the adversary E runs the protocol π for an input a received from the environment. Now a simulator $\mathsf{S} \in \mathcal{S}^{\mathsf{IT}}_{\mathsf{sh}}$ has to input some $a' \in \mathcal{X}_A$ to I_f independently of B's input b. Towards a contradiction assume $a' \neq a$ with non-negligible probability. Then, as the function f is redundancy-free, there are two possibilities:

$$\exists x_B \in \mathcal{X}_B \ : \ f_B(a, x_B) \neq f_B(a', x_B) \qquad \text{or} \tag{41}$$
$$\exists x_B, x'_B \in \mathcal{X}_B \ : \ f_A(a, x_B) \neq f_A(a, x'_B) \wedge f_A(a', x_B) = f_A(a', x'_B). \tag{42}$$

The first equation directly leads to distinguishable outputs for B, so the second equation must hold with non-negligible probability. Any adversary E under consideration runs the protocol π_A as we are in the semi-honest setting. We can wlog assume that E includes the protocol output y_A in its output, as a simulator that works for such an adversary E will also work if the protocol output y_A is not included. Then in the real execution we find $y_A = f_A(a,b)$ with overwhelming probability, in the ideal execution though, with a simulator that chooses b uniformly from $\{x_B, x'_B\}$ we find $y_A \neq f_A(a,b)$ with non-negligible probability as in the simulated setting the simulator receives $y'_A = f_A(a', x_B) = f_A(a', x'_B)$ with non-negligible probability, rendering the simulated y_A independent of b. This is in contradiction to the indistiguishability of $\mathsf{S}(\mathsf{E}) \circ I_f$ and $\mathsf{E} \circ \pi$.

So with overwhelming probability S will input a to to I_f. But then $\mathsf{S} \in \mathcal{S}^{\mathsf{IT}}_{\mathsf{pas}}$ already, and π is a passively secure protocol for I_f.

F Proof of Theorem 5.3

By Lem. 4.2 it is sufficient to show for redundancy-free functions f where $f = \hat{f}$ that $f \in \mathfrak{T}_{\mathsf{2act}} \iff f \in \mathfrak{T}'_{\mathsf{2act}}$. For all other (not redundancy-free) functions we then find

$$f \in \mathfrak{T}_{\mathsf{2act}} \overset{(4.2)}{\iff} \hat{f} \in \mathfrak{T}_{\mathsf{2act}} \iff \hat{f} \in \mathfrak{T}'_{\mathsf{2act}} \overset{(4.2)}{\iff} f \in \mathfrak{T}'_{\mathsf{2act}} \tag{43}$$

so $\mathfrak{T}_{\mathsf{2act}} = \mathfrak{T}'_{\mathsf{2act}}$ as claimed. So in the following let $f = \hat{f}$ be redundancy-free. We show $f \in \mathfrak{T}_{\mathsf{2act}} \iff f \in \mathfrak{T}'_{\mathsf{2act}}$ by proving the two implications $f \in \mathfrak{T}_{\mathsf{2act}} \implies f \in \mathfrak{T}'_{\mathsf{2act}}$ and $f \in \mathfrak{T}'_{\mathsf{2act}} \implies f \in \mathfrak{T}_{\mathsf{2act}}$ separately.

$f \in \mathfrak{T}'_{\mathsf{2act}} \implies f \in \mathfrak{T}_{\mathsf{2act}}$: This part of the proof is considerably simplified by use of the Symmetrization-Lemma of [KMQ08]. Said lemma states that any redundancy-free 2-party function f, which is not complete for 2-party computation, is locally mutually reducible to a symmetric function $\mathrm{sym}(f)$, i.e. $\mathrm{sym}(f)_A = \mathrm{sym}(f)_B$. Now we assumed $f = \hat{f}$ redundancy-free and $f \in \mathfrak{T}'_{\mathsf{2act}}$ implies $f \in \mathfrak{T}'_{\mathsf{2pas}} = \mathfrak{T}_{\mathsf{2pas}}$. As such f is not complete [Kus92, KMQ08] and we have $f \in \mathfrak{T}_{\mathsf{2act}} \iff \mathrm{sym}(f) \in \mathfrak{T}_{\mathsf{2act}}$, so wlog we may assume $f = \mathrm{sym}(f) = \mathrm{sym}(\hat{f})$.

We prove $f \in \mathfrak{T}'_{\mathsf{2act}} \implies f \in \mathfrak{T}_{\mathsf{2act}}$ by induction over the size of the input space $|\mathcal{X}_A \times \mathcal{X}_B|$. Now $f \in \mathfrak{T}'_{\mathsf{2act}}$ implies that f is either $f \in \mathfrak{T}_{\mathsf{2loc}}$ locally computable, in which case clearly $f \in \mathfrak{T}_{\mathsf{2act}}$ actively trivial and we are done, or f has a T-cut.

In the second case, by definition of the class $\mathfrak{T}'_{\mathsf{2act}}$ and wlog (else interchange A and B) the set $\mathcal{X}_B$ has a partition $\mathcal{X}_B^{(1)} \cup \mathcal{X}_B^{(2)} = \mathcal{X}_B$ such that

$$\begin{aligned}
&\forall\, x_A^{(1)}, x_A^{(2)} \in \mathcal{X}_A\ \exists x_A \in \mathcal{X}_A\ \forall\, x_B^{(1)} \in \mathcal{X}_B^{(1)}, x_B^{(2)} \in \mathcal{X}_B^{(2)} : \\
&\quad f_A(x_A^{(1)}, x_B^{(1)}) \neq f_A(x_A^{(1)}, x_B^{(2)}) \quad \wedge \\
&\quad f_B(x_A^{(1)}, x_B^{(1)}) = f_B(x_A, x_B^{(1)}) \quad \wedge \quad f_B(x_A^{(2)}, x_B^{(2)}) = f_B(x_A, x_B^{(2)}).
\end{aligned}$$

and the functions $f^{(1)} := f|_{\mathcal{X}_A \times \mathcal{X}_B^{(1)}}$, $f^{(2)} := f|_{\mathcal{X}_A \times \mathcal{X}_B^{(2)}}$ are $f^{(1)}, f^{(2)} \in \mathfrak{T}'_{\mathsf{2act}}$. By induction hypothesis then $f^{(1)}, f^{(2)} \in \mathfrak{T}_{\mathsf{2act}}$ as $|\mathcal{X}_A \times \mathcal{X}_B^{(i)}| < |\mathcal{X}_A \times \mathcal{X}_B|$ $(i \in \{1,2\})$. As such, there are protocols $\pi^{(1)}, \pi^{(2)}$ active PF securely implementing $I_{f^{(1)}}, I_{f^{(2)}}$ under simulators $\mathsf{S}^{(1)}, \mathsf{S}^{(2)}$.

We now construct a protocol π by defining a first protocol round where B sends A a message $m_1 \in \{1,2\}$ indicating whether B's input b is $b \in \mathcal{X}_B^{(1)}$ or $b \in \mathcal{X}_B^{(2)}$. If B sends no message, A just assumes $m_1 = 1$. Subsequently both parties proceed to run $\pi^{(m_1)}$.

It remains to prove this protocol π secure by providing a appropriate simulator S both for the case where A is corrupted and for the case where B is corrupted.

We begin with a corrupted A. S_A fixes the randomness of the adversary E (which it uses as a black-box) and feeds it the input x_{E} provided by the distinguisher D. Then the simulator S_A generates two copies $\mathsf{E}^{(1)}$ and $\mathsf{E}^{(2)}$, feeding $m_1 = 1$ to $\mathsf{E}^{(1)}$ and $m_1 = 2$ to $\mathsf{E}^{(2)}$, and runs $\mathsf{S}_A^{(1)}(\mathsf{E}^{(1)})$ and $\mathsf{S}_A^{(2)}(\mathsf{E}^{(2)})$ until these simulators produce inputs $a^{(1)}$ and $a^{(2)}$ for $I_{f^{(1)}}$ and $I_{f^{(2)}}$ respectively. Now by $f \in \mathfrak{T}'_{\textsf{2act}}$ there is an $a \in \mathcal{X}_A$ such that

$$\begin{aligned} \forall\, x_B^{(1)} \in \mathcal{X}_B^{(1)}, x_B^{(2)} \in \mathcal{X}_B^{(2)} :& \\ & f_A(a^{(1)}, x_B^{(1)}) \neq f_A(a^{(1)}, x_B^{(2)}) \quad \wedge \\ & f_B(a^{(1)}, x_B^{(1)}) = f_B(a, x_B^{(1)}) \quad \wedge \quad f_B(a^{(2)}, x_B^{(2)}) = f_B(a, x_B^{(2)}). \end{aligned}$$

The simulator S_A inputs this a to I_f and receives an output (recall that f symmetric, i.e. $f_A = f_B$)

$$y_A = f_A(a,b) = f_B(a,b) = \begin{cases} f_B(a^{(1)}, b) & \text{if } b \in \mathcal{X}_B^{(1)} \\ f_B(a^{(2)}, b) & \text{if } b \in \mathcal{X}_B^{(2)} \end{cases} = \begin{cases} f_A(a^{(1)}, b) & \text{if } b \in \mathcal{X}_B^{(1)} \\ f_A(a^{(2)}, b) & \text{if } b \in \mathcal{X}_B^{(2)} \end{cases} \tag{44}$$

Now if $y_A \in f_A(a, \mathcal{X}_B^{(1)})$ then S_A passes y_A to $\mathsf{S}_A^{(1)}(\mathsf{E}^{(1)})$ and returns the output of that subsimulator. Else if $y_A \in f_A(a, \mathcal{X}_B^{(2)})$ then S_A proceeds in the same way with $\mathsf{S}_A^{(2)}(\mathsf{E}^{(2)})$. The correctness of the simulation is obvious.

We now turn to a corrupted B. The simulator S_B fixes the randomness of the adversary E, passes it the input x_{E} provided by the distinguisher D, and runs the adversary E until E outputs a message m_1. If the adversary makes no output $m_1 \in \{1,2\}$ the simulator S_B takes $m_1 = 1$. Now the simulator S_B runs $\mathsf{S}_B^{(m_1)}(\mathsf{E})$ simply forwarding all inputs and outputs. This simulates the real execution, as depending on message m_1 the protocol $\pi^{(m_1)}$ is executed by the honest party A. The interaction with this protocol is then faithfully simulated by $\mathsf{S}_B^{(m_1)}$.

Note that we actually prove passive PF security above. The computation of locally computable functions is perfectly secure (induction base) and in the above argument if the subsimulators are perfectly secure, so are the resulting simulators (induction step).

$f \in \mathfrak{T}_{\textsf{2act}} \implies f \in \mathfrak{T}'_{\textsf{2act}}$: Recall that wlog $f = \hat{f}$ is redundancy-free. If $f \in \mathfrak{T}_{\textsf{2loc}}$ is locally computable then $f \in \mathfrak{T}'_{\textsf{2act}}$ and we are done. So consider an $f \notin \mathfrak{T}_{\textsf{2loc}}$. We show that f has a T-cut and then apply an inductive argument. As f is actively trivial we have $f \in \mathfrak{T}_{\textsf{2sh}}$ and $f \notin \mathfrak{T}_{\textsf{2loc}}$, so f has a K-cut. Wlog we have $\mathcal{X}_B = \mathcal{X}'_B \cup \mathcal{X}''_B$ and

$$\forall x_A \in \mathcal{X}_A \ : \ f_A(x_A, \mathcal{X}'_B) \cap f_A(x_A, \mathcal{X}''_B) = \emptyset \tag{45}$$

Now for any $x_A \in \mathcal{X}_A$ and regardless of the a priori distribution $\Pr_{X_A,X_B}$ on the inputs, as the function outputs on $\mathcal{X}'_B$ and $\mathcal{X}''_B$ differ, so must the protocol outputs y_A for A with overwhelming probability. As a result the statistical distance of the transcripts under input taken from $x_A, \mathcal{X}'_B$ or $x_A, \mathcal{X}''_B$ must be overwhelming as well:

$$\begin{aligned}
1 - \mathsf{negl} &< \Delta(\Pr_{Y_A|X_A,X_B}(\cdot, x_A, \mathcal{X}'_B), \Pr_{Y_A|X_A,X_B}(\cdot, x_A, \mathcal{X}''_B)) && (46)\\
&= \sum_{y_A \in \mathcal{Y}_A} |\Pr_{Y_A|X_A,X_B}(y_A, x_A, \mathcal{X}'_B) - \Pr_{Y_A|X_A,X_B}(y_A, x_A, \mathcal{X}''_B)| && (47)\\
&= \sum_{y_A \in \mathcal{Y}_A} |\sum_{t\in\Pi} \Pr_{Y_A,T|X_A,X_B}(y_A, t, x_A, \mathcal{X}'_B) - \sum_{t\in\Pi} \Pr_{Y_A,T|X_A,X_B}(y_A, t, x_A, \mathcal{X}''_B)| && (48)\\
&\leq \sum_{y_A \in \mathcal{Y}_A, t\in\Pi} |\Pr_{Y_A|T,X_A,X_B}(y_A, t, x_A, \mathcal{X}'_B)\Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}'_B) && (49)\\
&\qquad - \Pr_{Y_A|T,X_A,X_B}(y_A, t, x_A, \mathcal{X}''_B)\Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}''_B)|\\
&= \sum_{y_A \in \mathcal{Y}_A, t\in\Pi} |\Pr_{Y_A|T,X_A}(y_A, t, x_A)\Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}'_B) && (50)\\
&\qquad - \Pr_{Y_A|T,X_A}(y_A, t, x_A)\Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}''_B)|\\
&= \sum_{y_A \in \mathcal{Y}_A, t\in\Pi} \Pr_{Y_A|T,X_A}(y_A, t, x_A)\ |\Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}'_B) - \Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}''_B)| && (51)\\
&= \sum_{t\in\Pi} \underbrace{\sum_{y_A \in \mathcal{Y}_A} \Pr_{Y_A|T,X_A}(y_A, t, x_A)}_{=1} |\Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}'_B) - \Pr_{T|X_A,X_B}(t, x_A, \mathcal{X}''_B)| && (52)\\
&= \Delta(\Pr_{T|X_A,X_B}(\cdot, x_A, \mathcal{X}'_B), \Pr_{T|X_A,X_B}(\cdot, x_A, \mathcal{X}''_B)). && (53)
\end{aligned}$$

Note, that being independent from the priori distribution $\Pr_{X_A,X_B}$, this result applies to all subsets of $\mathcal{X}'_B$, $\mathcal{X}''_B$ as well. In particular for any $x'_B \in \mathcal{X}'_B$, $x''_B \in \mathcal{X}''_B$, $x_A \in \mathcal{X}_A$

$$\begin{aligned}
1 - \mathsf{negl} &< \Delta(\Pr_{Y_A|X_A,X_B}(\cdot, x_A, x'_B), \Pr_{Y_A|X_A,X_B}(\cdot, x_A, x''_B)) && (54)\\
&\leq \Delta(\Pr_{T|X_A,X_B}(\cdot, x_A, x'_B), \Pr_{T|X_A,X_B}(\cdot, x_A, x''_B)). && (55)
\end{aligned}$$

Now for any $\mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B$, $a \in \mathcal{X}_A$

$$\Delta(\Pr_{T_0|X_A,X_B}(\cdot, a, \mathcal{X}'_B), \Pr_{T_0|X_A,X_B}(\cdot, a, \mathcal{X}''_B)) = 0. \tag{56}$$

Hence considering all possible choices of values $x_A \in \mathcal{X}_A$ and cuts $\mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B$, as well as symmetrically $x_B \in \mathcal{X}_B$ and cuts $\mathcal{X}'_A \cup \mathcal{X}''_A = \mathcal{X}_A$, there must be a minimal round number r for which there is a distribution $\Pr_{X_A,X_B}$ on the inputs such that we have

$$\begin{aligned}
\alpha :=& \Delta(\Pr_{T_r|X_A,X_B}(\cdot, x_A, \mathcal{X}'_B), \Pr_{T_r|X_A,X_B}(\cdot, x_A, \mathcal{X}''_B)) > \mathsf{ntcbl} \qquad \text{or} && (57)\\
& \Delta(\Pr_{T_r|X_A,X_B}(\cdot, \mathcal{X}'_A, x_B), \Pr_{T_r|X_A,X_B}(\cdot, \mathcal{X}''_A, x_B)) > \mathsf{ntcbl} && (58)
\end{aligned}$$

Wlog we assume that the minimum r is achieved for some particular $x_A \in \mathcal{X}_A$ and a cut $\mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B$. Then, by minimality of the round number r, for all $r' < r$

$$\begin{aligned}
&\forall x_B \in \mathcal{X}_B,\ x_A \in \mathcal{X}_A, \mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B\ \mathcal{X}'_A \cup \mathcal{X}''_A = \mathcal{X}_A\ :\\
&\qquad \Delta(\Pr_{T_{r'}|X_A,X_B}(\cdot, x_A, \mathcal{X}'_B), \Pr_{T_{r'}|X_A,X_B}(\cdot, x_A, \mathcal{X}''_B)) < \mathsf{negl} \qquad \text{and} && (59)\\
&\qquad \Delta(\Pr_{T_r|X_A,X_B}(\cdot, \mathcal{X}'_A, x_B), \Pr_{T_r|X_A,X_B}(\cdot, \mathcal{X}''_A, x_B)) < \mathsf{negl}.
\end{aligned}$$

So the message m_r travels from B to A, as A cannot construct m_r for lack of information about x_B. So Eq. (59) and Eq. (57) hold for the given cut $\mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B$ and an *arbitrary* $x_A \in \mathcal{X}_A$, as m_r is constructed by B without information about x_A.

Furthermore note that we can find a cut $\mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B$ such that the above holds for every distribution $\Pr_{X_A,X_B}$ on the inputs. This can be seen using a hybrid argument over the inputs.

We need to show now

1. $\mathcal{X}_B' \cup \mathcal{X}_B'' = \mathcal{X}_B$ constitutes a K-cut
2. $\mathcal{X}_B' \cup \mathcal{X}_B'' = \mathcal{X}_B$ constitutes a T-cut
3. the argument perpetuates inductively

The cut $\mathcal{X}_B' \cup \mathcal{X}_B'' = \mathcal{X}_B$ constitutes a K-cut. Towards a contradiction assume we have $b' \in \mathcal{X}_B'$, $b'' \in \mathcal{X}_B''$, $a \in \mathcal{X}_A$ such that $f_A(a, b') = f_A(a, b'')$. Then for any simulated transcript T_S

$$\Pr_{T_\mathsf{S}|X_A,X_B}(\cdot, a, b') = \Pr_{T_\mathsf{S}|X_A,X_B}(\cdot, a, b'') \tag{60}$$

Now by security there exists a simulator S such that

$$\Delta(\Pr_{T_\mathsf{S}|X_A,X_B}(\cdot, a, b'), \Pr_{T|X_A,X_B}(\cdot, a, b')) < \mathsf{negl}, \tag{61}$$

$$\Delta(\Pr_{T_\mathsf{S}|X_A,X_B}(\cdot, a, b''), \Pr_{T|X_A,X_B}(\cdot, a, b'')) < \mathsf{negl}. \tag{62}$$

Then by transitivity

$$\Delta(\Pr_{T|X_A,X_B}(\cdot, a, b'), \Pr_{T|X_A,X_B}(\cdot, a, b'')) < \mathsf{negl} \tag{63}$$

which implies

$$\Delta(\Pr_{T_r|X_A,X_B}(\cdot, a, b'), \Pr_{T_r|X_A,X_B}(\cdot, a, b'')) < \mathsf{negl} \tag{64}$$

in contradiction to Eq. (57).

The cut $\mathcal{X}_B' \cup \mathcal{X}_B'' = \mathcal{X}_B$ is a T-cut. Consider an $a'' \in \mathcal{X}_A$ and the cut $\mathcal{X}_B' \cup \mathcal{X}_B'' = \mathcal{X}_B$ from Eq. (57). From Eq. (59) we have for any $a' \in \mathcal{X}_A$

$$\Delta(\Pr_{T_r|X_A,X_B}(\cdot, a', \mathcal{X}_B'), \Pr_{T_r|X_A,X_B}(\cdot, a'', \mathcal{X}_B')) < \mathsf{negl}. \tag{65}$$

Thus at round r the distribution of transcripts for different inputs of A differs still only negligibly. So with overwhelming probability a corrupted A can choose (uniformly among the matching ones) a new random string c_A' such that under the random string c_A' the input a' is consistent with the transcript T_r observed so far. The execution then proceeds according to π with the new values a' as input and c_A' as random string. We now fix an $a' \in \mathcal{X}_A$ and name the procedure just described π' and the induced transcript random variable T'.

We find that the distributions on the transcript T' (and thus also the output distributions) resulting from this procedure differ only negligibly from the transcript distributions obtained by initiating a normal protocol run with input a':

$$\Delta(\Pr_{T|X_A,X_B}(\cdot, a', \mathcal{X}_B'), \Pr_{T'|X_A,X_B}(\cdot, a'', \mathcal{X}_B')) \tag{66}$$

$$= \sum_{t\in\Pi} |\Pr_{T|X_A,X_B}(t, a', \mathcal{X}_B') - \Pr_{T'|X_A,X_B}(t, a'', \mathcal{X}_B')| \tag{67}$$

$$= \sum_{t\in\Pi} |\Pr_{T|T_r,X_A,X_B}(t, t_r, a', \mathcal{X}_B')\Pr_{T_r|X_A,X_B}(t, a', \mathcal{X}_B') - \Pr_{T'|T_r',X_A,X_B}(t, t_r, a'', \mathcal{X}_B')\Pr_{T_r'|X_A,X_B}(t, a'', \mathcal{X}_B')| \tag{68}$$

$$= \sum_{t\in\Pi} |\Pr_{T|T_r,X_A,X_B}(t, t_r, a', \mathcal{X}_B')\Pr_{T_r|X_A,X_B}(t, a', \mathcal{X}_B') - \Pr_{T|T_r,X_A,X_B}(t, t_r, a', \mathcal{X}_B')\Pr_{T_r|X_A,X_B}(t, a'', \mathcal{X}_B')| \tag{69}$$

$$= \sum_{t\in\Pi} \Pr_{T|T_r,X_A,X_B}(t, t_r, a', \mathcal{X}_B')|\Pr_{T_r|X_A,X_B}(t, a', \mathcal{X}_B') - \Pr_{T_r|X_A,X_B}(t, a'', \mathcal{X}_B')| \tag{70}$$

$$= \Delta(\Pr_{T_r|X_A,X_B}(\cdot, a', \mathcal{X}_B'), \Pr_{T_r|X_A,X_B}(\cdot, a'', \mathcal{X}_B')) < \mathsf{negl}. \tag{71}$$

We can proceed analogously for $\mathcal{X}_B''$ and thus have

$$\Delta(\mathrm{Pr}_{T|X_A,X_B}(\cdot, a', \mathcal{X}_B'), \mathrm{Pr}_{T'|X_A,X_B}(\cdot, a'', \mathcal{X}_B')) < \mathsf{negl}, \tag{72}$$

$$\Delta(\mathrm{Pr}_{T|X_A,X_B}(\cdot, a', \mathcal{X}_B''), \mathrm{Pr}_{T'|X_A,X_B}(\cdot, a'', \mathcal{X}_B'')) < \mathsf{negl}. \tag{73}$$

A corrupted A may then mount the following attack: A executes π with input a up to round r, obtaining a transcript t_r. Now if $\mathrm{Pr}_{T_r|X_A,X_B}(t_r, a'', \mathcal{X}_B') > \mathrm{Pr}_{T_r|X_A,X_B}(t_r, a'', \mathcal{X}_B'')$ (assuming uniform distribution on each set $\mathcal{X}_B'$ or $\mathcal{X}_B''$ of the partition) then A runs π' and π else. The resulting output distributions are indistinguishable (negligible statistical distance) from $\pi(a', X_B)$ and $\pi(a'', X_B)$ respectively as seen above. We designate this protocol execution by $\breve{\pi}$ and the induced random variable for the transcripts $\breve{T}$. Now take a distinguisher D that with probability $\frac{1}{2}$ each chooses X_B from $\mathcal{X}_B'$ or $\mathcal{X}_B''$ respectively with uniform distribution on each set $\mathcal{X}_B'$ or $\mathcal{X}_B''$ of the partition and provides A with no information.

Consider the conditional distribution of B's output under the given adverserial A and distinguisher D where we first take $b' \in \mathcal{X}_B'$ and then $b'' \in \mathcal{X}_B''$:

$$\begin{aligned}
\mathrm{Pr}_{Y_B|X_B}(y, b') &= \sum_{t\in\Pi} \mathrm{Pr}_{Y_B|\breve{T},X_B}(y, t, b') \mathrm{Pr}_{\breve{T}|X_B}(t, b') && (74)\\
&= \sum_{t\in\Pi} \mathrm{Pr}_{Y_B|T,X_B}(y, t, b') \mathrm{Pr}_{\breve{T}|X_B}(t, b') && (75)\\
&\overset{(59)}{\approx} \sum_{t\in\Pi} \mathrm{Pr}_{Y_B|T,X_B}(y, t, b') \left(\frac{1+\alpha}{2} \mathrm{Pr}_{T|X_A,X_B}(t, a', b') + \frac{1-\alpha}{2} \mathrm{Pr}_{T|X_A,X_B}(t, a'', b') \right) && (76)\\
&\approx \frac{1+\alpha}{2} \mathrm{Pr}_{Y_B|X_A,X_B}(y, a', b') + \frac{1-\alpha}{2} \mathrm{Pr}_{Y_B|X_A,X_B}(y, a'', b') && (77)\\
&\approx \frac{1+\alpha}{2} 1_{y=f_B(a',b')} + \frac{1-\alpha}{2} 1_{y=f_B(a'',b')}; && (78)\\
\mathrm{Pr}_{Y_B|X_B}(y, b'') &= \sum_{t\in\Pi} \mathrm{Pr}_{Y_B|\breve{T},X_B}(y, t, b'') \mathrm{Pr}_{\breve{T}|X_B}(t, b'') && (79)\\
&= \sum_{t\in\Pi} \mathrm{Pr}_{Y_B|T,X_B}(y, t, b'') \mathrm{Pr}_{\breve{T}|X_B}(t, b'') && (80)\\
&\overset{(59)}{\approx} \sum_{t\in\Pi} \mathrm{Pr}_{Y_B|T,X_B}(y, t, b'') \left(\frac{1-\alpha}{2} \mathrm{Pr}_{T|X_A,X_B}(t, a', b'') + \frac{1+\alpha}{2} \mathrm{Pr}_{T|X_A,X_B}(t, a'', b'') \right) && (81)\\
&\approx \frac{1-\alpha}{2} \mathrm{Pr}_{Y_B|X_A,X_B}(y, a', b'') + \frac{1+\alpha}{2} \mathrm{Pr}_{Y_B|X_A,X_B}(y, a'', b'') && (82)\\
&\approx \frac{1-\alpha}{2} 1_{y=f_B(a',b'')} + \frac{1+\alpha}{2} 1_{y=f_B(a'',b'')}. && (83)
\end{aligned}$$

In the ideal setting and for the distinguisher D as described above, the input a of A as provided to the ideal functionality by the simulator is independent of the input b of B. Thus we find for the conditional distribution of the output Y_B^I of B in the ideal case, both for $b \in \mathcal{X}_B'$ and $b \in \mathcal{X}_B''$

$$\mathrm{Pr}_{Y_B^I|X_B}(y, b) = \sum_{a\in\mathcal{X}_A \,:\, y=f_B(a,b)} \mathrm{Pr}_{X_A}(a) \tag{84}$$

where the simulator can only adjust $\mathrm{Pr}_{X_A}(a)$.

By security of the protocol (simulatability) we must have a *single* distribution Pr_{X_A} such that

$$\mathsf{negl} > \Delta(\mathrm{Pr}_{Y_B|X_B}(y, b'), \mathrm{Pr}_{Y_B^I|X_B}(y, b')) \tag{85}$$

$$\mathsf{negl} > \Delta(\mathrm{Pr}_{Y_B|X_B}(y, b''), \mathrm{Pr}_{Y_B^I|X_B}(y, b'')) \tag{86}$$

We can conclude that there is an $a \in \mathcal{X}_A$ such that $\Pr_{X_A}(a) > \mathsf{ntcbl}$ and $f_B(a,b') = f_B(a',b')$, $f_B(a,b'') = f_B(a'',b'')$ for all $b' \in \mathcal{X}'_B$ and $b'' \in \mathcal{X}''_B$:

$$\begin{aligned}
\mathsf{negl} &> \Delta(\Pr_{Y_B|X_B}(y,b'), \Pr_{Y_B^I|X_B}(y,b')) && (87)\\
&= \sum_{y\in\mathcal{Y}_B} |\Pr_{Y_B|X_B}(y,b') - \Pr_{Y_B^I|X_B}(y,b')| && (88)\\
&\approx \sum_{y\in\mathcal{Y}_B} |\frac{1+\alpha}{2}1_{y=f(a',b')} + \frac{1-\alpha}{2}1_{y=f_B(a'',b')} - \sum_{a\in\mathcal{X}_A\,:\,y=f_B(a,b')} \Pr_{X_A}(a)| && (89)\\
&= \sum_{y\in\mathcal{Y}_B} |\frac{1+\alpha}{2}1_{y=f(a',b')} + \frac{1-\alpha}{2}1_{y=f_B(a'',b')} - \sum_{a\in\mathcal{X}_A} \Pr_{X_A}(a)1_{y=f_B(a,b')}|. && (90)
\end{aligned}$$

If $f_B|_{\mathcal{X}_A\times\mathcal{X}'_B}$ or $f_B|_{\mathcal{X}_A\times\mathcal{X}''_B}$ are constant, then f already has a T-cut. So wlog we consider $a', a'' \in \mathcal{X}_A$ such that there is a $b' \in \mathcal{X}'_B$ and a $b'' \in \mathcal{X}''_B$ where $f_B(a',b') \neq f_B(a'',b')$ and $f_B(a',b'') \neq f_B(a'',b'')$. But then the obvious solution $\Pr_{X_A}(a') \approx \frac{1+\alpha}{2}$, $\Pr_{X_A}(a'') \approx \frac{1-\alpha}{2}$, $\Pr_{X_A}(a) \approx 0$ for $a' \neq a \neq a''$ leads to a contradiction with

$$\begin{aligned}
\mathsf{negl} &> \Delta(\Pr_{Y_B|X_B}(y,b''), \Pr_{Y_B^I|X_B}(y,b'')) && (91)\\
&\approx \sum_{y\in\mathcal{Y}_B} |\frac{1-\alpha}{2}1_{y=f(a',b'')} + \frac{1+\alpha}{2}1_{y=f_B(a'',b'')} - \sum_{a\in\mathcal{X}_A} \Pr_{X_A}(a)1_{y=f_B(a,b'')}|. && (92)
\end{aligned}$$

So there must be an $a \in \mathcal{X}_A$ where $\Pr_{X_A}(a) > \mathsf{ntcbl}$. But then also $f(a,b') = f(a',b')$, $f(a,b'') = f(a'',b'')$. In fact, if there is only one such a, we find

$$\begin{aligned}
\Pr_{X_A}(a) &\approx \alpha, && (93)\\
\Pr_{X_A}(a') &\approx \frac{1-\alpha}{2}, && (94)\\
\Pr_{X_A}(a'') &\approx \frac{1-\alpha}{2}. && (95)
\end{aligned}$$

Extending the Argument Finally we need to show, that the argument presented above applies inductively until we are left with locally computable restrictions of the original function.

To this end we consider $f' = f|_{\mathcal{X}_A\times\mathcal{X}'_B}$ and $f'' = f|_{\mathcal{X}_A\times\mathcal{X}''_B}$. Unfortunately we cannot readily conclude that $f \in \mathfrak{T}_{\mathsf{2act}}$ implies $f', f'' \in \mathfrak{T}_{\mathsf{2act}}$. Take for instance f' and some corrupted B. Then S simulates correctly for I_f by security of π for f. On $I_{f'}$ however the simulation may fail, as S may rely on making an input $b \in \mathcal{X}''_B$ to I_f, which will be rejected by $I_{f'}$.

However, for our proof above we employ only a specific class $\mathcal{E}$ of adversaries, namely semi-honest adversaries and adversaries that generally adhere to the protocol π, but possibly attempt to "switch" their input. We show that security against this class $\mathcal{E}$ of adversaries already implies $f \in \mathfrak{T}'_{\mathsf{2act}}$, so in particular $f \in \mathfrak{T}_{\mathsf{2act}} \implies f \in \mathfrak{T}'_{\mathsf{2act}}$. So it is sufficient to argue that both f' and f'' are indeed in the class $\mathfrak{T}''_{\mathsf{2act}}$ of functions secure against this specific subclass $\mathcal{E}$ of adversaries. Then we can proceed inductively or simply argue by contraposition: Take an $f \in \mathfrak{T}''_{\mathsf{2act}}$, $f \notin \mathfrak{T}'_{\mathsf{2act}}$ where $|\mathcal{X}_A \times \mathcal{X}_B|$ minimal. We have shown that such an f must have a T-cut decomposing f into f' and f''. Now $f', f'' \in \mathfrak{T}''_{\mathsf{2act}}$ (this we show below) and hence by minimality of f we have $f', f'' \in \mathfrak{T}'_{\mathsf{2act}}$. But then $f' \in \mathfrak{T}'_{\mathsf{2act}}$ by definition of $\mathfrak{T}'_{\mathsf{2act}}$ in contradiction to the choice of f. We hence find

$$f \in \mathfrak{T}_{\mathsf{2act}} \implies f \in \mathfrak{T}''_{\mathsf{2act}} \implies f \in \mathfrak{T}'_{\mathsf{2act}}. \qquad (96)$$

So we need to show that f' and f'' are indeed in the class $\mathfrak{T}''_{\mathsf{2act}}$ of functions secure against the subclass of adversaries $\mathcal{E}$ used in the proof above. Now clearly for corrupted A we have f' and f'' as secure as f, since the same simulators can be applied. For a corrupted B consider adversaries and distinguishers as in the argument above where simply the set $\mathcal{X}_A$ is replaced by $\mathcal{X}'_B$ and the set $\mathcal{X}_B$ by $\mathcal{X}_A$. Then the outcome Y of the real protocol execution will be $Y \in f(\mathcal{X}_A, \mathcal{X}'_B)$ with overwhelming probability. Now $\mathcal{X}'_B \cup \mathcal{X}''_B = \mathcal{X}_B$ is a K-cut, meaning

$$\forall x_A \in \mathcal{X}_A \ : \ f(x_A, \mathcal{X}'_B) \cap f(x_A, \mathcal{X}''_B) = \emptyset. \tag{97}$$

But then, as $\mathsf{S}(B)$ is a valid simulator for I_f under π, $\mathsf{S}(B)$ must with overwhelming probability give an input $b \in \mathcal{X}'_B$ to I_f. Hence $\mathsf{S}(B)$ is actually a valid simulator for $I_{f'}$ with respect to π. In other words $f' \in \mathfrak{T}''_{\mathsf{2act}}$.

G Proofsketch of Theorem 6.1

We first show $\mathfrak{T}_{\mathsf{2qu}} \neq \mathfrak{T}_{\mathsf{2sh}}$. Due to the impossibility of implementing unbiased coin flipping (Kitaev, quoted in [ABDR04]) it is impossible to have XOR$\in \mathfrak{T}_{\mathsf{2qu}}$ (which is sufficient to implement coin flipping), however XOR is in $\mathfrak{T}_{\mathsf{2sh}}$.

Next we sketch how to prove $\mathfrak{T}_{\mathsf{2qu}} \subseteq \mathfrak{T}_{\mathsf{2sh}}$. Towards a contradiction assume there is a function $f \in \mathfrak{T}_{\mathsf{2qu}} \setminus \mathfrak{T}_{\mathsf{2sh}}$. Then there is an effcient secure quantum protocol π for f, but no classical protocol secure against a semi-honest adversary. We can assume f to be minimal in the size of the input space $|\mathcal{X}_A \times \mathcal{X}_B|$. Then f must already be redundancy-free and hence, according to Lemma 4.3 we have $f \notin \mathfrak{T}_{\mathsf{2pas}}$. This implies that f does not have a K-cut.

The secure quantum protocol π runs in a polynomial number of rounds, and we can assume the number of rounds to be padded to $p(k)$ rounds independent of the inputs.

The proof will be by induction over the number of rounds, analogously to the proof of Lemma 3.4. However we will have to consider measurements on the quantum data.

For round number r of a protocol π and a quantum non-demolition measurement[9] M which measures just one bit we define an adversary $\mathsf{E}_{r,M}$. This adversary honestly follows the protocol π until round r. In round r (before sending or after receiving a message) the adversary $\mathsf{E}_{r,M}$ performs the measurement M. Then the adversary continues to honestly follow the protocol, however, his quantum data might be disturbed by the measurement M and we have to see that this will remain undetected by the uncorrupted party with noticeable probability:

A quantum state before a quantum non-demolition measurement yielding one bit of output and the quantum state after the measurement have a noticeable overlap and the probability for any measurement to give identical results for the two states is noticeable. This noticeable probability upper bounds the probability that the uncorrupted party will detect any difference between the behaviour of $\mathsf{E}_{r,M}$ and honest behaviour.

Let the inputs x_A, x_B for the parties be chosen uniformly at random. According to the above discussion of the probability with which the measurement performed by the adversary remains undetected we have that for every input x_A, x_B the protocol π will not abort (i.e. terminate successfully) with noticeable probability. We will next prove by induction that for these non-aborting protocol runs it holds that the bit measured by the adversary is statistically independent of the input of the uncorrupted party. This will then lead to the desired contradiction.

In round $r = 0$ before any message was sent any bit resulting from a measurement performed by the adversary is statistically independent of the input of the uncorrupted party.

Now we take as induction hypothesis that no statistical difference could be observed by any adversary up to round r. In round $r+1$ a message is sent. If this message is sent by the adversary then still no statistical difference can be detected, hence we only need to consider the case where the message in round $r+1$ is sent by the uncorrupted party to the adversary (for notational convenience we assume in the following that B is the corrupted party). We assume that there is an adversary $\mathsf{E}_{r+1,M}$ who can after receiving the message

[9]I.e. a textbook measurement which disturbs the quantum state just as much as required by the laws of quantum mechanics.

observe a statistical difference with respect to different inputs of the uncorrupted party. There is one party which, if corrupted, would be able to observe the statistical difference first with a probability $\geq 1/2$ and we can thus conclude that with a probability $\geq 1/2$ the result of any measurement which could be performed on the quantum data of the uncorrupted party is statistically independent of the input of the adversary (otherwise we just change the roles of the parties). Hence the (noticeable) probability of observing this statistical difference in round $r+1$ is almost independent of the input of the adversary $\mathsf{E}_{r+1,M}$. Due to this statistical difference the set of inputs $\mathcal{X}_A$ of the uncorrupted party can be partitioned into two disjoint subsets $\mathcal{X}'_A, \mathcal{X}''_A$ ($\mathcal{X}_A = \mathcal{X}'_A \cup \mathcal{X}''_A$) such that the conditional probability of any input in $\mathcal{X}'_A$ is noticeably higher than the conditional probability of any input in $\mathcal{X}''_A$ given the measured bit. Since the function f has no K-cut there exist inputs $x'_A \in \mathcal{X}'_A$ and $x''_A \in \mathcal{X}''_A$ for the uncorrupted party and an input x_B for the adversary such that $f_B(x'_A, x_B) = f_B(x''_A, x_B)$. So with noticeable probability the protocol will not abort and the adversary will have a noticeable advantage in distinguishing x'_A and x''_A while using input x_B. This is impossible in an ideal model secure function evaluation of f. We can conclude that for every adversary $\mathsf{E}_{r,M}$ and every round r the bit measured by the adversary is statistically independent of the input of the uncorrupted party.

However, there is an adversary who is looking at his output in the last round (and otherwise runs the protocol). This adversary will be able to see a statistical difference with respect to the input of the uncorrupted party or the function is locally computable. This contradicts the statistical independence of every measured bit and concludes the proof.

H Proofsketch of Theorem 7.1

A LT secure protocol is already by definition secure against semi-honest adversaries.

Conversely, let $f \in \mathfrak{T}_{\mathsf{2sh}}$. We show that $f \in \mathfrak{T}_{\mathsf{2lts}}$ by constructing a LT secure protocol for f. First, we obtain an unconditionally hiding commitment scheme from the given OWF [HR07]. Then the parties commit to their inputs and give a zero-knowledge argument of knowledge of their inputs. Now the parties execute the semi-honestly secure protocol π_f for f, that exists as $f \in \mathfrak{T}_{\mathsf{2sh}}$ with the following modifications: After each computation step they commit to the result and give a perfect ZK argument that it was computed correctly. Instead of sending messages they now simply open commitments.

This procedure (computationally) binds the parties to correct protocol execution, (computationally) forcing semi-honest behavior (or abort). On the other hand no additional information is flowing compared to the original protocol π_f, due to the unconditional hiding property of the commitment scheme and the perfect ZK property of the arguments. Therefore the information theoretic security of the protocol against semi-honest adversaries is maintained. In particular the security can not be violated anymore after protocol termination, even by unbounded adversaries.

A simulator as demanded by the definition of security is easily constructed from the above. The proofs of knowledge in the beginning of the protocol are used to extract an input. A simulator can then pass this input to the ideal functionality and obtain output from the ideal functionality. The simulator then determines an input of the honest party that is consistent with the input and output extracted from the adversary respectively received from ideal functionality. Using this input the simulator can simulate a regular protocol execution toward the adversary and use the output of the adversary as its own. Indistiguishability of such a simulation follows once again directly from the above arguments.

I Proofsketch of Lemma 7.2

We show the two implications separately:

CO-OT+ $\Longrightarrow$ LTS-DA Let $f \in \mathfrak{T}_{\mathsf{2sh}}$ be a SH trivial two party function. From Theorem 4.3 we obtain a SH secure protocol $\widetilde{\pi}$. We "pad" the protocol with dummy-messages, such that the number of rounds does not depend on the inputs anymore. We view the resulting protocol as a distributed circuit, where the gates are owned by A or B, depending on which party executes a specific computation. Communication is then nothing else than a wire between gates owned by different parties.

From this distributed circuit we can obtain a computationally secure protocol for the function f using the Goldreich compiler [Gol04], that (computationally) securely computes any circuit using a CO-OT+ functionality.

As in the case of plain LT security each party first commits to their input using unconditionally hiding bit-commitments and proofs knowledge of the input using a perfectly zero-knowledge argument of knowledge. However, in the following, since we are interested in an LT secure protocol π for f with designated aborter A, we need to take special care when applying these transformations. Concretely, we only use unconditionally hiding bit-commitments and perfect zero-knowledge arguments and we take care to apply the CO-OT+ in such a fashion that it is LT secure against B when computing a gate owned by A and LT secure against A when computing a gate owned by B. When the protocol is complete, the result is first opened towards A, who opens the result to B if it receives the inputs ok and terminates otherwise.

As shown in [Gol04] the protocol π computes f computationally securely and the designated abort property is provided through the opening procedure, since as Goldreich shows no party learns anything until the opening phase. It remains to argue the long-term security.

In π the structure of the underlying passively IT secure protocol $\widetilde{\pi}$ is preserved. Due to the way we employ CO-OT+, informationtheoretically, each party only receives information about inputs and outputs of gates it computes itself in $\widetilde{\pi}$. Therefore no party receives more information than in $\widetilde{\pi}$, and after termination of the protocol security, even against unbounded distinguishers, is guaranteed.

Simulators as demanded by the definition of security are easily constructed from the above. The proofs of knowledge in the beginning of the protocol are used to extract an input. A simulator can then pass this input to the ideal functionality and obtain output from the ideal functionality. The simulator then determines an input of the honest party that is consistent with the input and output extracted from the adversary respectively received from ideal functionality. Using this input the simulator can simulate a regular protocol execution toward the adversary and use the output of the adversary as its own. Indistiguishability of such a simulation follows once again directly from the above arguments.

LTS-DA $\implies$ CO-OT+ Let $x_A, x_{B,0}, x_{B,1} \in \{0,1\}$ and consider the (symmetric) function

$$f(x_A, x_{B,0}, x_{B,1}) = \Big(\big((1 \oplus x_A)x_{B,0} \oplus x_A x_{B,1}, x_A\big), \big((1 \oplus x_A)x_{B,0} \oplus x_A x_{B,1}, x_A\big)\Big). \tag{98}$$

A SH secure protocol for computing f is as follows: A sends x_A to B (x_A is part of the output anyway). B computes $f(x_A, x_{B,0}, x_{B,1})$ and sends the result to A, so $f \in \mathfrak{T}_{\mathsf{2sh}}$.

Now assume there is a protocol π computing f that is LTS-DA. We define the protocol $\widetilde{\pi}$ as the protocol π where A always inputs $o = 0$ (aborts) and B hence makes no output. We first show that $\widetilde{\pi}$ is a computationally secure OT protocol.

To this end, let $\mathsf{S}(\mathsf{E})$ be a simulator such that $\Delta_{\mathsf{D}}(\pi, \mathsf{S}(\mathsf{E}) \circ I_f^{\mathsf{des}})$ negligible for all $\mathsf{D} \in \mathcal{D}^{\mathsf{CO}}$. Such a simulator S exists by definition of LTS-DA. The same simulator S works for OT.

Corrupted B: Assume that $\Delta_{D'}(\widetilde{\pi}, \mathsf{S}(\mathsf{E}) \circ \mathrm{OT})$ non-negligible. We now construct a distinguisher D, that employs the same E and runs D', but additionally has A input $o = 0$. Note that in both scenarios the view of E remains identical, as does the information available to the simulator S. So, in contradiction to our assumption D has non-negligible advantage.

Corrupted A: Assume that $\Delta_{\mathsf{D}'}(\widetilde{\pi}, \mathsf{S}(\mathsf{E}) \circ \mathrm{OT})$ non-negligible. We now construct a distinguisher D, that employs the same E and runs D' but suppresses the output of B by setting input $o = 0$ for A. Note that in both scenarios the view of E remains identical, as does the information available to the simulator S. So, in contradiction to our assumption D has non-negligible advantage.

It remains to show that $\widetilde{\pi}$ is an OT protocol LT secure against A, so let A be corrupted. Now, by definition of LTS-DA there is a simulator $\mathsf{S}(\mathsf{E})$ such that $\Delta_{\mathsf{D}}(\mathsf{E} \circ \pi', \mathsf{S}(\mathsf{E}) \circ I_f^{\mathsf{ab}})$ is negligible for all distinguishers $\mathsf{D} \in \mathcal{D}^{\mathsf{LT}} = \mathsf{Algo}$ and where π' is π with $o = 1$ fixed (A does not abort). Now we apply this simulator S to OT, simply ignoring the abort flag o. Assume there is a computationally bounded adversary E and a computationally unbounded distinguisher D such that the advantage $\Delta_{\mathsf{D}'}(\widetilde{\pi}, \mathsf{S}(\mathsf{E}) \circ \mathrm{OT})$ is non-negligible. Note that in both scenarios the view of E remains identical, as does the information available to the simulator S. So, in contradiction to our assumption D has non-negligible advantage $\Delta_{\mathsf{D}}(\mathsf{E} \circ \pi', \mathsf{S}(\mathsf{E}) \circ I_f^{\mathsf{ab}})$.

J Classification of 2-party Functions

Definition J.1 (Consistent Renaming [KMQ08]). A function $f^{(1)} \in \mathfrak{F}_2$ is a consistent renaming of a function $f^{(2)} \in \mathfrak{F}_2$ iff there are

1. a bijective map $\sigma_A : \mathcal{X}_A^{(1)} \to \mathcal{X}_A^{(2)}$,
2. a bijective map $\sigma_B : \mathcal{X}_B^{(1)} \to \mathcal{X}_B^{(2)}$,
3. $\forall a \in \mathcal{X}_A^{(1)}$ a bijective map $\sigma_a : f_A^{(1)}(a, \mathcal{X}_B^{(1)}) \to f_A^{(2)}(\sigma_A(a), \mathcal{X}_B^{(2)})$,
4. $\forall b \in \mathcal{X}_B^{(1)}$ a bijective map $\sigma_b : f_B^{(1)}(\mathcal{X}_A^{(1)}, b) \to f_B^{(2)}(\mathcal{X}_A^{(2)}, \sigma_B(b))$.

We then say $f^{(1)}$ and $f^{(2)}$ are renamings: $f^{(1)} \equiv f^{(2)}$. ◊

Note that in Theorem 8.3 we actually have

$$\begin{aligned}
\mathfrak{T}_{\mathsf{2act}} \setminus \mathfrak{T}_{\mathsf{2pas}} &= \{f \in \mathfrak{T}_{\mathsf{2act}} \mid \bar{f} \not\equiv \hat{f}\} \\
\mathfrak{T}_{\mathsf{2sh}} \setminus \mathfrak{T}_{\mathsf{2pas}} &= \{f \in \mathfrak{T}_{\mathsf{2sh}} \mid \bar{f} \not\equiv \hat{f}\} \\
\mathfrak{F}_{\mathsf{2pas}}^{\mathsf{nct}} &= \{f \in \mathfrak{F}_{\mathsf{2sh}}^{\mathsf{nct}} \mid \bar{f} \equiv \hat{f}\} \\
\mathfrak{F}_{\mathsf{2act}}^{\mathsf{nct}} &= \mathfrak{F}_{\mathsf{2sh}}^{\mathsf{nct}} \cup (\mathfrak{T}_{\mathsf{2sh}} \setminus \mathfrak{T}_{\mathsf{2act}}) \\
\mathfrak{C}_{\mathsf{2pas}} &= \mathfrak{C}_{\mathsf{2act}} \cup \{f \in \mathfrak{F}_2 \mid \hat{f} \not\equiv \bar{f}\}.
\end{aligned}$$

Using the examples in Fig. 1 we can illustrate that the inclusions of Theorem 8.3 are indeed proper:

$$f^{(1)} \in \mathfrak{T}_{\mathsf{2act}} \setminus \mathfrak{T}_{\mathsf{2pas}} \subsetneq \mathfrak{C}_{\mathsf{2pas}} \setminus \mathfrak{C}_{\mathsf{2act}} \tag{99}$$

$$f^{(2)} \in \mathfrak{T}_{\mathsf{2pas}} \setminus \mathfrak{T}_{\mathsf{2act}} \subsetneq \mathfrak{F}_{\mathsf{2act}}^{\mathsf{nct}} \setminus \mathfrak{F}_{\mathsf{2sh}}^{\mathsf{nct}} \tag{100}$$

$$f^{(3)} \in \mathfrak{T}_{\mathsf{2sh}} \setminus \mathfrak{T}_{\mathsf{2act}} \cup \mathfrak{T}_{\mathsf{2pas}} \tag{101}$$

$$f^{(4)} \in \mathfrak{C}_{\mathsf{2pas}} \cap \mathfrak{F}_{\mathsf{2sh}}^{\mathsf{nct}} \subsetneq \mathfrak{C}_{\mathsf{2pas}} \setminus \mathfrak{C}_{\mathsf{2act}}. \tag{102}$$