Privacy-Enhancing First-Price Auctions Using Rational Cryptography

Peter Bro Miltersen Jesper Buus Nielsen Nikos Triandopoulos

Department of Computer Science
University of Aarhus, Denmark
{bromille, buus, nikos} (at) cs . au . dk

Abstract

We consider enhancing a sealed-bid single-item auction with *privacy* concerns, our assumption being that bidders primarily care about monetary payoff and secondarily worry about exposing information about their type to other players and learning information about other players' types. To treat privacy explicitly within the game theoretic context, we put forward a novel *hybrid utility* model that considers both fiscal and privacy components in the players' payoffs.

We show how to use rational cryptography to approximately implement a given *ex interim* individually strictly rational equilibrium of such an auction (or any game with a winner) without a trusted mediator through a cryptographic protocol that uses only point-to-point authenticated channels between the players. By "ex interim individually strictly rational" we mean that, given its type and before making its move, each player has a strictly positive expected utility, i.e., it becomes the winner of the auction with positive probability. By "approximately implement" we mean that, under cryptographic assumptions, running the protocol is a computational Nash equilibrium with a payoff profile negligibly close to the original equilibrium.

In addition the protocol has the stronger property that no collusion, of any size, can obtain more by deviating in the implementation than by deviating in the ideal mediated setting which the mechanism was designed in. Also, despite the non-symmetric payoffs profile, the protocol always correctly terminates.

Contents

1 Introduction

1.1 The problem: realizing privacy-enhanced auctions

Consider the following scenario: A *seller* S wants to sell an item to one of n *bidders* $P_1, P_2, \ldots, P_n$ using a sealed bid auction, e.g., a first-price or a second-price (Vickrey) auction. To optimize their expected payoff in these settings, the bidders P_i are to submit their true valuation of the item (e.g., in a Vickrey auction) or more generally a function of their true valuation (e.g., the Bayesian equilibrium strategy in a first-price auction) as their bid. However, in the scenario we suggest, matters are complicated by the following issues: First, bidders are not happy revealing any information related to their true valuation to the seller. Second, bidders would also be unhappy if other buyers gain information about their valuation. On the other hand, they would appreciate learning something about the valuations of the other players if they get the chance.

Of course, such concerns can be handled by assuming the availability of a trusted *mediator* M to whom bidders do not mind revealing their valuations. Such a trusted party can collect the bids, determine the winner, and ensure that the seller and the winning bidder get in touch with one another.

The problem we address in this paper is what to do when a trusted mediator is not available. A well known fact is that cryptography and in particular secure multiparty computation can sometimes be used to replace a trusted mediator. In particular, we may consider the following scheme for replacing M:

1. The seller commits by contract in advance to sell the item to the first bidder P_i that can present a document digitally signed by all other buyers, stating that P_i is the buyer who should get the item. The document should also specify at which price P_i is to get the item (in case S has a reservation price, the contract can state that the document is only valid if this reservation price is met).

2. The bidders perform a secure multiparty computation that simulates the mediator of the mediated auction and produces such a signed document containing the correct winner-price pair.

Indeed, previous papers concerned with secure cryptographic implementations of auctions have suggested schemes along these lines, e.g., [16, 18]. Also, at least in one instance such a scheme (for a double auction, not a first-price or Vickrey auction) has been implemented in practice [2].

However, there are issues that make this not quite solve our problem. The introduced privacy concerns of the bidders dictate the use of joint computations that eventually produce non-symmetric outputs for the bidders, where specifically only the winner gets the winning contract while other bidders learn nothing. In this setting, nothing enforces the winner to carry out the transaction with S. This destroys the standard equilibrium analysis of a Vickrey auction which crucially depends on the winner being forced to buy, to make it costly to bid higher than ones valuation. This suggests using a first-price auction instead, but even then it is not obvious that rational parties will carry out the protocol outlined above when they have privacy concerns. In general, we wish to extend classical equilibrium analysis of auctions of game theory to cryptographic auction protocols and make an argument that a rational party *has no incentive to deviate from following the protocol as specified*. A concrete problem is *protocol participation*. In realizations of games with non-symmetric final payoffs (like auctions), an agent has no incentive to continue and complete the protocol as soon as he realizes that he cannot be the winner. In contrast, the traditional analysis of multiparty computation assumes that at least *some* parties are "honest" and will carry out all steps of the protocol, no matter what (Bradford *et al.* [3] study the problem of protocol-completion incentives that exist in an auction when participants realize that they cannot win the auction, but in a model where privacy is not captured in players' rationality). Recent papers on rational cryptography have analyzed multiparty computation as a game [1, 6, 8, 10, 11, 15, 17, 19] but, aiming at simultaneous information exchange and modeling rationality through pure information loss/gain, these works cannot precisely model auctions with non-symmetric outcomes/payoffs and a setting where utilites are a mix of monetary utilities and privacy concerns.

Matters are complicated by the fact that even the mediated auction *does* leak some information (e.g, the mere fact that a bidder did not win gives him information about the winning bid). Hence, it is intuitively clear that if the privacy has high weight, existing equilibria in the classical case are disturbed (e.g., truth telling is no longer even a Nash equilibrium for Vickrey auctions), and for a high enough emphasis on privacy, not taking part in the auction (say, by submitting the bid 0, independent of the valuation) becomes a strictly dominant strategy. Whatever analysis one obtains will have to be consistent with this fact.

Perhaps the biggest challenge, finally, is to design a protocol as the above in a way that can be realized using *today's Internet computing and communication machinery*. While there are recent results that allow removing mediators in very general classes of games [9, 12, 13], these works use communication channels such as simultaneous broadcast (like most of existing works on rational cryptography) or physical envelopes that are quite restrictive or even unobtainable when considering a practical implementation over the Internet.

1.2 Outline of our contribution

In this paper, we suggest a rational cryptographic protocol for replacing a trusted mediator in a sealed-bid single-item auction. The protocol uses only point-to-point authenticated channels between the buyers and can therefore be implemented on the Internet. As described above, the assumption is that the seller, in advance commits to selling the item to the bidder who presents a certain document digitally signed by other bidders. That is, we assume that such a bit string has monetary value for exactly one of the bidders. Furthermore, all bidders in advance commit to buying the item if the seller presents a signed contract with the bidders name on it. Besides such monetary concerns, we have to assign utilities to players so that the privacy concerns outlined in the previous section are adequately modeled. But because of the monetary value of the signed document, we deviate from previous works on secure auction implementation where privacy was treated at a second-phase technical level *outside* of the scope of game and parties' strategies, but also from previous works in rational cryptography where utilities were *solely* concerned with gain or exposure of information. Instead, we propose a *hybrid utility model* where agents are interested in both monetary gain from participating in the auction as well as in maintaining the privacy of their type (e.g., valuation). Their actual utility is a linear combination of a *monetary utility* and an *information utility*. For the information utility, rather than postulating one particular utility measure, we set up axioms which encompass a wide variety of privacy concerns. We note that a different hybrid utility model is studied by Halpern and Pass [7].

We consider a general class of single-item sealed-bid auctions in the standard Bayesian setup of auction theory (see next section for formal definition) and *without* privacy concerns. We formally define the corresponding mediated game *with* privacy concerns, as modeled using our hybrid utilities. In general, as we indicated in an intuitive way in the previous subsection, if high weight is put on the information part of the hybrid utilities, then the equilibria of the privacy-aware game may be very different from the equilibria of the original game. However, in a variant of the first-price auction with discrete valuations and bids, we observe that when the weight put on the information concern is "much smaller" than the weight on the monetary concern (i.e., when agents are assumed to be *greedy then paranoid*), then the original auction mechanism (with a small twist) is an equilibrium of the mediated game.

To accurately model auctions with privacy concerns but without the participation of the seller (or generally any privacy-enhanced mechanism with one winner and non-symmetric outcomes), we introduce *mediation with reject*, a slightly relaxed idealized setting where the winner is merely given the choice to reject the winning contract. This explicitly captures a crucial characteristic of any Internet-based implementation of the auction mechanisms we study: at some specific point in the computation, the winner (and only himself) will locally compute the winning contract (similar to the *revelation point* of [10, 11]); nobody prevents him from turning off his machine. Therefore, mediation with reject offers the proper abstraction for studying auction mechanisms. As we will see, this reject option can drastically affect the existence of equilibria.

Our main result is the following. We can, under a mild assumption stated below, relate a given equilibrium (suggested behavior) π of the mediated game (in the setting of mediation with reject) to a corresponding suggested behavior π' of our unmediated cryptographic protocol so that π' has the same payoff profile as π, up to a negligible amount, and for computationally bounded agents that care "much less" about privacy than about money, following π' is an ϵ-Nash equilibrium where ϵ is negligible. Here, "negligible" is defined relative to the strength of the cryptography used.

The assumption we need is the following: The equilibrium π should have *ex interim* strictly positive expected utility for all players. That is, after a player learns his type but before he makes his move, his expected conditional utility is strictly positive. As an example, our protocol enables a variant of the first-price auction and the corresponding Bayesian bidding equilibrium to be conducted by computationally bounded, rational but *not* necessarily honest buyers over the Internet in a realistic way, along the lines suggested in the previous section, without a trusted mediator and without participation of the seller. Also, despite the non-symmetric payoffs profile, our protocol always correctly terminates, under the assumption above.

We remark that while Kol and Naor [10] identify ϵ-Nash equilibrium as a minimum rationality requirement for rational cryptography, a body of work [1, 6, 8, 15], and in particular Kol and Naor [10, 11], suggests using stronger notions of solution concepts and equilibria that are *not susceptible to backwards inductions*. While these stronger notions are indeed desirable, our protocol as stated does not satisfy them. We take the standpoint that even ϵ-Nash is a meaningful property. Stronger notions are appropriate when we think of the recommended (equilibrium) strategy as something an agent is to *consciously* follow. On the other hand, if we think of the recommended strategy as *software* that we as protocol designers recommend agents to use, establishing that using this software is ϵ-Nash is evidence that agents will not bother hacking or replacing it.

Sketch of the protocol. The idea behind our protocol is intuitive and quite simple. Given individual signing keys and corresponding (publicly known) verification keys for some signature scheme, and also their private bids, the agents engage a randomized joint computation during which the winner (highest bidder) obtains a digital contract that contains the winner-price pair (j, p), signed by all agents. Conceptually, the protocol is divided in a fixed (and large) number E of stages, called *epochs*. Sequentially during each epoch e, each agent i receives a value $V_{e,i}$ and thus has the opportunity to obtain the winning contract, if he is the winner: the contract is released to the winner during one, randomly chosen epoch $e_0 \in [E]$ (with probability 2^{-e} in epoch $e = 1, \ldots, E-1$), whereas all other received values (by party $i \neq j$ at epoch e_0 or by any party at all other epochs) are set to a special nil value. That is, $V_{e,i}$ is nil, except $V_{e_0,j} = \mathsf{sign}(j, p)$. This exact randomized functionality is implemented by first using secure multiparty computation, at the end of which each party k obtains an *additive* share of each value $V_{e,i}$ (or $\perp$ if agents provide invalid inputs). From this point on, the E epochs of the protocol are realized sequentially, by simply asking in a round-robin fashion each agent to send its share of $V_{e,i}$ to agent i, and repeat for all $i = 1, \ldots, n$.

Crucially for the protocol's stability, agent i is asked to refuse to send his shares in subsequent value-reconstruction transmissions, as soon as he experiences denial to reconstruct his own value $V_{e,i}$. This implies that with positive probability any agent deviating at epoch $e < E$ destroys his winning possibility in a later epoch, given that the winning-reconstruction epoch e_0 is hidden in the computation; this does not hold in epoch E, but $e_0 = E$ occurs only with negligible probability, so the protocol is an ϵ-Nash for a negligible ϵ.

Organization of the paper. In Section 2 we provide a brief description of the classical auctions model in the (pure) mediated setting. In Section 3 we introduce a definitional framework for protocol games implementing privacy-enhanced auctions, where we describe in details our circuit-based communication and computational model, our hybrid utility model and our rationality model. In Section 4 we present the mediated setting with reject and show the existence of privacy-enhanced Nash equilibria for first-price auctions and their preservation in the mediated with reject setting. In Section 5 we introduce privacy-enhanced Nash implementation, our core proof technique for designing and showing privacy-enhanced Nash equilibria in a modular manner. Finally, in Section 6 we present our concrete protocol for realizing auctions over the Internet. Complete technical proofs are presented in the Appendix.

2 Classical Auctions

First, we recap the classical (i.e., privacy-oblivious) model of a sealed-bid single-item auctions as an incomplete information, Bayesian, game. Such a game is played by parties (bidders) $P_1, P_2, \ldots, P_n$ competing for an item to be sold. The game starts with each bidder P_i receiving a private *type* $t_i \in T_i$ where T_i is the *type space* of the bidder. The Bayesian setup indicates that the vector $(t_1, t_2, \ldots, t_n)$ is drawn at random from a commonly known distribution on $T = T_1 \times T_2 \ldots \times T_n$. This distribution is known as the *common prior* and will also be denoted by T. The valuation of bidder i for the item to be sold is given by $v_i = v_i(t_1, t_2, \ldots, t_n)$. In the simplest and most common case (the case of private values), $v_i(t_1, t_2, \ldots, t_n) = t_i$. Based on his type, Bidder i strategically chooses and submits a *bid* b_i. That is, a *strategy* of party i is given by a map B_i mapping types to bids. Based on the bids $b = (b_1, b_2, \ldots, b_n)$ and possibly a random source, an *allocation mechanism* Mec now allocates the item to a single bidder j (the *winner*) and computes a *price* p. We write $(j, p) \leftarrow \mathrm{Mec}(b)$. The *monetary utility* of bidder j is $r_j = g(v_j, p)$ for some function g, most commonly $r_j = v_j - p$ (this is the case for a risk neutral agent P_j as he gets the item at price p and values it v_j), while the payoff of other bidders are $r_i = 0$ (as they do not get the item and do not have to pay anything). For a first-price auction, j is the bidder with the highest bid (with ties broken at random), and p is the bid of the winner. For the case of the Vickrey auction, j is the bidder with the highest bid, while p is the highest bid if the bid of the winner is removed. A Bayes-Nash (for brevity, from now on just Nash) equilibrium for the auction is a (possibly randomized) bidding strategy for each bidder that maximizes his expected payoff, assuming other bidders follow their prescribed strategy.

3 Protocol Games

To enhance the classical auction with privacy concerns, we have to explicitly model privacy as part of the utility function and consider appropriate notions of equilibria. For this we in turn have to explicitly model the communication of the protocol, and the information collected by a party during the protocol execution.

3.1 Communication and protocol execution

We start with a formal communication and protocol execution model. It is convenient to use a unified model, which allows to capture both the mediated setting and the Internet-like setting using the same formalism, which we will call a *communication device*. To be able to use cryptography, we also want to model the fact that parties are computationally bounded to get the desired definitions; this we do by simply restricting the strategy space to poly-time strategies. The model we present in this section is not specific for auctions.

Communication devices. A protocol is of the form $\pi = (\pi_1, \dots, \pi_n)$, where π_i is a program describing the strategy of party P_i. These programs communicate in rounds using a communication device $\mathcal{C}$. In each round, $\mathcal{C}$ takes an input $m_i \in \{0,1\}^d$ from each π_i and outputs a value $o_i \in \{0,1\}^d$ to each π_i. I.e., in each round, $\mathcal{C}$ is a function $(\{0,1\}^d)^n \to (\{0,1\}^d)^n, (m_1, \dots, m_n) \mapsto (o_1, \dots, o_n)$. Which function is computed might depend on the inputs and outputs of previous rounds and the randomness of $\mathcal{C}$.

Parties and strategies. We let the strategy π_i for each party P_i be an interactive circuit for R rounds. The circuit consists of $1+R$ circuits $\pi_i^{(0)}, \pi_i^{(1)}, \dots, \pi_i^{(R)}$. The circuit $\pi_i^{(0)}$ takes $a+b$ bits as input and outputs $a+b$ bits, where a, b are integers specified by the circuit. In each round π_i takes as input a *state* $s \in \{0,1\}^a$, and a *message* $m \in \{0,1\}^b$ (from the communication device $\mathcal{C}$). The output of the circuit is parsed as an updated state $s' \in \{0,1\}^a$ and a message $m' \in \{0,1\}^b$ (for the communication device). In the first round, the state consists of a uniformly random bits and the message is the type of P_i. In subsequent rounds, s is the updated state s' from the previous round and m is the value sent by $\mathcal{C}$ for that round.

Because we consider in subsequent settings protocols using cryptography, we actually do not consider a single fixed circuit π_i. Rather π_i specifies a circuit $\pi_i(\kappa)$ for each value κ of the security parameter.[1] Each $\pi_i(\kappa)$ is allowed to have different state and message lengths $a(\kappa), b(\kappa)$. Similarly we let $\mathcal{C}$ specify a communication device $\mathcal{C}(\kappa)$ for each $\kappa \in \mathbb{N}$. Also, for technical reasons we adopt a non-uniform model, where the sequence of strategies $\pi_i(1), \pi_i(2), \dots$ need not have a finite description using e.g. a Turing machine.[2] For a function $\tau : \mathbb{N} \to \mathbb{N}$ we use Π^τ to denote the circuit families π_i where for all κ the size of $\pi_i(\kappa)$ is at most $\tau(\kappa)$. A strategy space Π^τ is always defined in context of some communication device $\mathcal{C}$ which for each κ expects (and produces) messages of some fixed size $d(\kappa) \in \mathbb{N}$. We require that Π^τ only contains circuit families where $b(\kappa) = d(\kappa)$ for all κ.

Executions. Let $\mathcal{C}$ be some communication device, let $\pi = (\pi_1, \dots, \pi_n)$ be a protocol, where $\pi_i \in \Pi^\tau$, and let T be a distribution on types. An *execution* proceeds as in Fig. 1. We call $o = (o_1, \dots, o_n) = (o_1^{(R)}, \dots, o_n^{(R)})$ the *outcome* of the protocol. I.e., the outcome is the last round of outputs from $\mathcal{C}$. We call the output of the last circuit $w_i = (s_i^{(R+1)}, m_i^{(R+1)})$ the *local output* of party P_i, and call $w = (w_1, \dots, w_n)$ the *local outputs*. We use $(t, o, w) \leftarrow (\pi, \mathcal{C})(T)$ to denote the distribution of (t, o, w) on a *random execution*, i.e., for uniformly random ρ, random $t \leftarrow T$ and uniform randomness of $\mathcal{C}$.

[1]The value of κ determines the key lengths of the underlying cryptographic primitives, e.g., cryptosystems, signature schemes.

[2]Insisting on π_i having a uniform description might make it impossible to analyze the games for different values of κ independently, or would at least require an explicit argument that this can be done: Changing the strategies $\pi_i(\kappa)$ for some values of the security parameter κ might necessitate a change for other values to ensure that the sequence $\pi_1(1), \pi_1(2), \dots$ still has a uniform description. The utility of changing strategy for one specific game (i.e., for a fixed value of κ) might therefore not be possible without considering the utility of changing strategy at other security levels, which seems un-intuitive and might unnecessarily complicate analysis. Adopting a non-uniform model deals with such concerns in a straight-forward manner.

1. Sample $(t_1, \ldots, t_n) \leftarrow T$, and sample uniformly random $\rho_i \in \{0,1\}^a$ for $i = 1, \ldots, n$.
2. For $i = 1, \ldots, n$, run $\pi_i^{(0)}$ on (ρ_i, t_i) to produce $(s_i^{(1)}, m_i^{(1)})$. Then for $r = 1, 2, \ldots, R$, do the following: First run $\mathcal{C}$ on $(m_1^{(r)}, \ldots, m_n^{(r)})$ to produce $(o_1^{(r)}, \ldots, o_n^{(r)})$, and then, for $i = 1, \ldots, n$, run $\pi_i^{(r)}$ on $(s_i^{(r)}, o_i^{(r)})$ to produce $(s_i^{(r+1)}, m_i^{(r+1)})$.

Figure 1: An execution

Utilities. The *utility* of P_i is a real valued function u_i. We assume that u_i is a function of the types, the outcomes and the local outputs. We use u to denote $(u_1, \ldots, u_n)$. We sometimes use $u_i(T, \mathcal{C}, \pi)$ to denote the *expected utility* of P_i, i.e., $u_i(T, \mathcal{C}, \pi)$ is the expected value of $u_i(t, o, w)$ for $(t, o, w) \leftarrow (\pi, \mathcal{C})(T)$.

3.2 The mediator and the Internet as communication devices

It is easy to see that the mediator can be expressed as a communication device (see also Appendix A).

Internet-like communication. For analyzing protocols for Internet-like networks we need an explicit communication device $\mathcal{C}_{\texttt{int}}$ modeling communication on the Internet. Ideally we want $\mathcal{C}_{\texttt{int}}$ to closely reflect how messages are delivered on the Internet and similar networks. Since our results are very robust with respect to the exact specification of $\mathcal{C}_{\texttt{int}}$ we will, however, use a rather idealized device.

We assume that the device can deliver secure messages directly between each pair of parties. This can be achieved using standard Internet technology by e.g. establishing SSL connections between each pair of parties. Using such a model we avoid the introduction of unnecessary complications, like the exact structure of the network used to carry the messages. On the other hand, we do not want the simplification of $\mathcal{C}_{\texttt{int}}$ to make the model unrealistic. One issue which we explicitly want to avoid is using a communication device $\mathcal{C}_{\texttt{int}}$ which allows simultaneous message exchange. Hence we adopt a minimally complicated $\mathcal{C}_{\texttt{int}}$ by simply assuming that in each round, one predefined party receives messages from all other parties.

A communication device $\mathcal{C}_{\texttt{int}}^{\text{gen,Out}}$ works as follows:

1. In round 1, sample a key pair $(pk_i, sk_i) \leftarrow \text{gen}(1^\kappa)$ for each P_i and output $((pk_1, \ldots, pk_n), sk_j)$ to P_j.
2. In rounds $r = 2, \ldots, R-1$ the input from each party P_i is parsed as a message $m_i \in \{0,1\}^k$ for some fixed k. The output to $P_{r \bmod n}$ is $(m_1, \ldots, m_n)$. The output to all other parties is `silence`.
3. In round $r = R$, compute $(o_1, \ldots, o_n) = \text{Out}(msg)$, where msg are all messages sent in the previous rounds, and output the outcome o_i to P_i.

Figure 2: An Internet-Like Device $\mathcal{C}_{\texttt{int}}^{\text{gen,Out}}$

Finally we assume the existence of a PKI. We model this in a simplistic manner by letting the device distribute the keys. In the last round the device will define an output by the last round of messages output to the parties. We assume that this is a function Out of the messages sent. The details are given in Fig. 2.

3.3 Information and monetary utilities

Information utilities. We now turn our attention to the valuation of the information collected and leaked during the execution of the protocol. For this we use the local outputs.

We let the local output w_i capture the type information collected by P_i. I.e., if P_i wants to take some type information with it from the execution, it outputs it as part of w_i. We assume that P_i valuates the type information collected using an *information utility* $q_i(t, w)$. Note that q_i can measure information collected by P_i as well as by other parties: maybe $q_i(t, w) = 1$ if $w_i = t_1$ but $q_i(t, w) = -1$ if $w_1 = t_i$.

We allow q_i to express arbitrary privacy concerns, except for two restrictions, described now. To ensure that q_i is consistent with the view of knowledge from cryptography, that knowledge is the information which can be computed in poly-time, we require that q_i is poly-time computable. We also need that q_i valuates the collection of knowledge non-negatively. Let $(w_1, \ldots, w_n)$ be any distribution and let $(w'_1, \ldots, w'_n)$ be the distribution where $w'_i = \perp$ and $w'_{-i} = w_{-i}$. Then we require that $q_i(t, (w'_1, \ldots, w'_n)) \leq q_i(t, (w_1, \ldots, w_n)) + \epsilon$, where ϵ is negligible. Here $\perp$ is some fixed symbol, playing the role of no information. In words: loosing the output w_i, and all other things being equal, cannot be valuated as significantly positive by P_i. We say that q_i *prefers collection of knowledge*.

Definition 1 *We call q_i an* admissible privacy measure *if it is poly-time and prefers collection of knowledge.*

Our protocols will work only for privacy measures which are sufficiently small compared to the expected utility of playing the game. So it is convenient to have the following measure of the privacy concerns' size.

Definition 2 *For an information utility $q_i(t, w)$ we call $\|q_i\| = (\max_{t,w} q_i(t, w)) - (\min_{t,w} q_i(t, w))$ both the* weight of the information utility *and the* weight of the privacy concern.

We will not be concerned about exactly how the q_i measures privacy concerns, as we are going to develop protocols that are ϵ-Nash *for all admissible measures* $q = (q_1, \ldots, q_n)$ with sufficiently small weight compared to the expected monetary utility.

Monetary utilities. Complementing the information utility we have the notion of a *monetary utility*, which is just a utility function $r_i(t, o)$ which depends only on the types and the outcomes. For generality we allow r_i to change with κ. We do, however, assume that the absolute value of r_i is bounded by a polynomial in κ. The intuitive reason for this assumption is that we need to use cryptography, which withstands only poly-time attacks. In concrete terms, if you use a protocol where it would cost \$10000000 to buy enough computing power to break the cryptography, do not use it to play a game where anyone can win \$10000001. In this light, bounding the monetary utility by a polynomial can be seen as a extremely crude way to deal with the price of computation in the utility function.

We design mechanisms which work only if the expected monetary utility of the parties is large compared to how they valuate information. We define a measure of this. For any t_i occurring with non-negligible probability as component i in $(t_1, \ldots, t_n) \leftarrow T$, let $(t, o, w) \leftarrow (\pi, \mathcal{C})(T)_{t_i}$ denote the conditional distribution of $(t, o, w) \leftarrow (\pi, \mathcal{C})(T)$ given that the i'th component of t is t_i, and let I_i denote the expected value of $u_i(t, o, w)$ for $(t, o, w) \leftarrow (\pi, \mathcal{C})(T)_{t_i}$. We call I_i the *ex interim* expected utility of P_i for t_i, which is just its expected utility after seeing type t_i. For a given security level κ we let $\gamma(\kappa)$ be the minimum over all parties P_i and all t_i (occurring with non-negligible probability) of the *ex interim* expected utility of P_i given t_i.

Definition 3 *We call $\gamma : \mathbb{N} \to \mathbb{R}$, as defined above, the ex interim* rationality *of* $(\pi, \mathcal{C}, T)$.

3.4 Privacy-enhanced Nash equilibrium

When we design a mechanism, we can control the monetary utility $r_i(t, o, w) = r_i(t, o)$. For instance, we simply enforce $r_i(t, o, w) = t_i - p$ for the winner of a Vickrey auction. In principle parties can have arbitrary utilities $u_i(t, o, w)$, even if running a protocol with the purpose of implementing some mechanism. We will, however, only consider settings where the part of the utility which cannot be explained as monetary utility from the designed mechanisms can be explained by an admissible measure of privacy. I.e., we assume that $q_i(t, o, w) = u_i(t, o, w) - r_i(r, o)$ is an admissible measure of privacy. In particular, $q_i(t, o, w) = q_i(t, w)$ and

$$u_i(t, o, w) = r_i(t, o) + q_i(t, w) \ .$$

For the later schemes involving cryptography, we follow Kol and Naor [10] who argued that ϵ-Nash equilibrium for negligible ϵ is the appropriate minimum rationality requirement for "information games".

Definition 4 *For a single protocol π (i.e., for a fixed κ), a strategy space Π^τ, a distribution T on types, and $\epsilon \in \mathbb{R}$, $\epsilon > 0$, we call π an ϵ-Nash equilibrium (for $\mathcal{C}, T, \Pi^\tau$) if it holds for all parties P_i and all $\pi_i^* \in \Pi^\tau$ that $u_i((\pi_i^*, \pi_{-i}), \mathcal{C}, T) - u_i(\pi, \mathcal{C}, T) \leq \epsilon$. For a protocol π (specified for all κ), strategy space Π^τ, a distribution T on types, we call π a* computational Nash equilibrium *(for $\mathcal{C}, T, \Pi^\tau$) if for all polynomials τ there exists a negligible ϵ such that $\pi(\kappa)$ is an $\epsilon(\kappa)$-Nash equilibrium (for $\mathcal{C}, T, \Pi^{\tau(\kappa)}$) for all κ.*

Our notion of computational Nash is technically slightly different from the original notion introduced by Dodis *et al.* [4]. The notion is, however, similar enough that we fell that we can soundly reuse the terminology of a computational Nash equilibrium. As already mentioned, implementations of monetary mechanisms can only be expected to work if the weight of the privacy concerns is relatively small. We thus capture the size of the information utility as part of the definition of a privacy-enhanced Nash equilibrium.

Definition 5 *Fix a monetary utility r and a privacy weight α. We call a protocol a* privacy-enhanced Nash equilibrium *(for r and α) if it is a computational Nash equilibrium for $u = r + q$ for all admissible privacy measures q with $\|q\| \leq \alpha$.*

In words, a privacy-enhanced Nash equilibrium has the property that no matter how the parties valuate information (as long as it has weight at most α), there is no deviation which will allow any party to learn more valuable information, unless such a deviation would have it lose an equivalent amount of monetary utility. This implies that there is no way a party P_j can efficiently extract knowledge from its view of the protocol extra to that of its local output w_j. If there was, it could do so and output this extra knowledge, which would make some q_i prefer this. Therefore the recommended local outputs of a privacy-enhanced mechanism precisely specify what information each party can collect; not as an explicit requirement, but because we use computational Nash equilibrium as solution concept.

We extend the above notion to cover also collusions of size t. In Definition 4 we consider $C \subset \{1, \dots, n\}$ with $|C| \leq t$ and we consider deviations π_C^* consisting of π_i^* for $i \in C$. We call π t-resilient if $u_i((\pi_C^*, \pi_{-C}), \mathcal{C}, T) - u_i(\pi, \mathcal{C}, T) \leq \epsilon$ for all $i \in C$. I.e., for all collusion of size t and all possible deviations, not even a single party in the collusion gets extra utility. This directly defines notions of *t-resilient computational Nash equilibrium* and *t-resilient privacy-enhanced Nash equilibrium*.

As a concrete example of a privacy-enhanced Nash equilibrium for an auction mechanism with standard mediation, we consider a single-item, sealed-bid, first-price auction with three bidders and independent private valuations, each distributed uniformly in $\{1, 3\}$. The bidding space is the natural numbers, including 0. A general theory of equilibria of first-price auctions with integral valuations and bids is the topic of a recent paper by Escamocher *et al.* [5]. For the special case at hand, it is straightforward to check that the symmetric profile $\pi = (B_1, B_2, B_3)$, with $B_1 = B_2 = B_3 = B$, where $B(1) = 0$ and $B(3) = 1$, is a Nash equilibrium of the classical (privacy oblivious) auction. The *ex interim* expected payoff for a bidder with valuation 1 is $1/12$. The ex interim expected payoff for a bidder with valuation 3 is $7/6$. Since these numbers are strictly bigger than 0, it is easy to check that for any privacy measure with sufficiently small weight, the equilibrium persists.

4 Mediation with Reject

Towards designing a protocol that implements an auction on an Internet-like network without the participation of the seller and that is a privacy-enhanced Nash equilibrium, we first study reasonable, privacy-enhanced Nash equilibria for a highly idealized setting that better fits the real-world setting. The idealized setting that we consider is called *mediation with reject*, a slightly weakening of the normal mediated setting.

The communication device $\mathcal{C}_{\text{REJ}}^{\text{Mec}}$ is parameterized by a number of rounds R and works as follows:

compute result: In round 1, take the input b_i from each P_i, let $b = (b_1, \ldots, b_n)$, sample $(j, p) \leftarrow \text{Mec}(b)$, and let $o_j = (j, p)$ and $o_i = \texttt{sorry}$ for $i \neq j$.

offer contract: The contract is offered as follows:

1. Output o_i to each P_i.
2. For each P_i with $o_i \neq \texttt{sorry}$, if P_i does not input `accept` before round R, then set $o_i \leftarrow \texttt{sorry}$.

define outcome: In the last round, R, output o_i to each P_i.

side-channel: In rounds $r = 2, \ldots, R-1$ the device, in addition, allows point-to-point communication as in $\mathcal{C}_{\texttt{int}}$.[a]

The recommend strategy $\pi_j^{\texttt{rej}}$ for each P_j is to input $b_j \leftarrow B_j(t_j)$, to input `accept` in round 2, and to give the local output $w_j = (t_j, o_j)$.

[a]Not used by the mechanism, this point-to-point communication allows collusions to have side communication.

Figure 3: The Mediated Setting with Reject $(\mathcal{C}_{\text{REJ}}^{\text{Mec}}, \pi_B^{\texttt{rej}})$ for mechanism $\text{Mec} = (B_1, \ldots, B_n, \text{Mec}, r)$

Mediation with reject is defined only for mechanisms with one winner. A mechanism (B, Mec, r) is said to have *one winner* if it holds for all possible outputs $(o_1, \ldots, o_n) = \text{Mec}(b)$ that there exists P_i such that $o_i \neq \texttt{sorry}$ and $o_j = \texttt{sorry}$ for $P_j \neq P_i$ and that $t_j(t, o, w) = 0$ when $o_j = \texttt{sorry}$. So, only one party has monetary utility from the mechanism. In the *mediated setting with reject* the parties are first given the outcome of the mechanism as an *offered outcome* of the game. The winner (the P_i with $o_i \neq \texttt{sorry}$) then has the offer to change its outcome of the game to $o_i = \texttt{sorry}$, in which case it will receive monetary utility $r_i(t, o, w) = 0$, consistent with the rule that $r_j(t, o, w) = 0$ when $o_j = \texttt{sorry}$. For such a mechanism we can assume that Mec simply outputs the index j of the winner and some extra value p which we call the *price*. Details are given in Fig. 3.

Privacy-enhanced Nash equilibria for first-price auctions with standard mediation exist for certain settings of the parameters, as exemplified above. By the next lemma (proof in Appendix B), they are preserved in the setting with mediation with reject.

Lemma 1 *If π^{med} is a (ϵ-)Nash equilibrium for $(\mathcal{C}^{med}_{\mathrm{Mec}}, u, T)$, and $\alpha \leq \gamma$, then π^{rej} is a (ϵ-)Nash equilibrium for $(\mathcal{C}^{\mathrm{Mec}}_{\mathrm{REJ}}, u, T)$.*

On the other hand, it is easy to check that the standard truth telling equilibrium of a second-price (Vickrey) auction is in general *not* a privacy-enhanced Nash equilibrium in the setting of mediation with reject. The fact that the winner is not forced to make the transaction makes bidding infinity (or the highest possible bid) a dominant strategy. For non-trivial privacy concerns, this dominant strategy is also a strictly better reply than truth telling to a strategy profile where the other bidders bid truthfully. Thus, mediation with reject is a setting where we observe a *separation* between first-price and second-price auctions with respect to the existence of reasonable privacy-enhanced Nash equilibria, fully justifying the importance of this abstraction.

5 Nash Implementation and Hybrid Proofs

When working with protocol games using cryptography it is convenient to have a framework which allows for modular proofs, where e.g. cryptographic primitives are introduced one by one. This is testified by the widespread use of hybrid proofs in the cryptographic literature. We suggest a notion of Nash-preserving implementation which gives exactly such a framework.

Consider an idealized communication device $\mathcal{C}^{\mathtt{ide}}$, as e.g. $\mathcal{C}^{\mathrm{Mec}}_{\mathrm{REJ}}$ and a recommended protocol $\pi^{\mathtt{ide}}$ for $\mathcal{C}^{\mathtt{ide}}$. Consider then a more real-life communication device $\mathcal{C}^{\mathtt{imp}}$, like a communication device modeling an Internet-like network, and consider a protocol $\pi^{\mathtt{imp}}$ for that communication device. We will consider $(\mathcal{C}^{\mathtt{imp}}, \pi^{\mathtt{imp}})$ an implementation of $(\mathcal{C}^{\mathtt{med}}, \pi^{\mathtt{med}})$ in the spirit of Nash if the parties do not get more incentives to deviate when they interact in $(\mathcal{C}^{\mathtt{imp}}, \pi^{\mathtt{imp}})$ than when they interact in $(\mathcal{C}^{\mathtt{med}}, \pi^{\mathtt{med}})$. Again, since we want to allow the use of cryptography, we allow for a negligible small slack.

Definition 6 (privacy-enhanced Nash implementation) *Fix a distribution T on types and a monetary utility $r = (r_1, \ldots, r_n)$. Let $(\mathcal{C}^{imp}, \pi^{imp})$ and $(\mathcal{C}^{ide}, \pi^{ide})$ be two settings. We say that $(\mathcal{C}^{imp}, \pi^{imp})$ is a t-*resilient privacy-enhanced Nash implementation *of $(\mathcal{C}^{ide}, \pi^{ide})$ if for all $u = r + q$, where $q = (q_1, \ldots, q_n)$ are admissible measures of privacy with weight at most α, there exists a negligible ϵ such that:*

No less utility: *For all P_l, $u_l(T, \mathcal{C}^{imp}, \pi^{imp}) \geq u_l(T, \mathcal{C}^{ide}, \pi^{ide}) - \epsilon$.*

No more incentive to deviate: *For all $C \subset \{1, \ldots, n\}$, $|C| \leq t$ and all strategies $\pi^{imp^*}_C$ for $\mathcal{C}^{imp}$, there exists a strategy $\pi^{ide^*}_C$ for $\mathcal{C}^{ide}$ such that $u_l(T, \mathcal{C}^{ide}, (\pi^{ide^*}_C, \pi^{ide}_{-C})) \geq u_l(T, \mathcal{C}^{imp}, (\pi^{imp^*}_C, \pi^{imp}_{-C})) - \epsilon$ for all $P_l \in C$.*

It is straightforward to verify the following theorem. For completeness the proof appears in Appendix C.

Theorem 1 *For fixed T and r, it holds for all settings $(\mathcal{C}, \pi)$, $(\mathcal{D}, \gamma)$ and $(\mathcal{E}, \delta)$ that:*

Privacy-enhanced Nash preservation: *If $(\mathcal{C}, \pi)$ is a t-resilient privacy-enhanced Nash implementation of $(\mathcal{D}, \gamma)$ and γ is a t-resilient privacy-enhanced Nash equilibrium for $\mathcal{D}$, then π is a t-resilient privacy-enhanced Nash equilibrium for $\mathcal{C}$ with a utility profile negligibly close to that of $(\mathcal{C}, \gamma)$, i.e., $|u_l(T, \mathcal{C}, \pi) - u_l(T, \mathcal{D}, \gamma)|$ is negligible for all P_l and for all considered $u = r + q$.*

Transitivity: *If $(\mathcal{C}, \pi)$ is a t-resilient privacy-enhanced Nash implementation of $(\mathcal{D}, \gamma)$ and $(\mathcal{D}, \gamma)$ is a t-resilient privacy-enhanced Nash implementation of $(\mathcal{E}, \delta)$, then $(\mathcal{C}, \pi)$ is a t-resilient privacy-enhanced Nash implementation of $(\mathcal{E}, \delta)$.*

The above theorem suggests a design principle for designing privacy-enhanced mechanisms for complicated setting, like Internet-like networks: First design a desirable privacy-enhanced Nash equilibrium for a highly idealized setting, like the mediated setting. This bares relatively little weight extra to that of traditional mechanism design. Then do a privacy-enhanced Nash implementation of the designed mechanism for the complicated setting, and appeal to privacy-enhanced Nash preservation. In proving that the implementation is a privacy-enhanced Nash implementation, transitivity can allow for modular proofs.

6 Rational Auctions for Internet-Like Networks

In this section we present a protocol for running an auction on an Internet-like network. The overall goal is get a protocol which is a privacy-enhanced Nash and which implements a reasonable utility profile. The design methodology will be to consider any privacy-enhanced Nash equilibrium in the mediated setting with reject and then provide a privacy-enhanced Nash preserving implementation of this equilibrium in the Internet-like network. Our privacy-enhanced Nash preserving implementation is generic, and applies to essentially all mechanisms with one winner (defined in Section 4). For technical reasons, we first consider an intermediate, more elaborate, communication device which will serve as middle ground between the mediated setting with reject and the unmediated one.

The communication device $\mathcal{C}^{\mathtt{unf}}_{\text{Mec}}$ is parameterized by a number of rounds R and works as follows:

compute result: In round 1 each P_i inputs (b_i, l_i, f_i), where b_i is a bid, $f_i \in [0, \frac{1}{2}]$ and $l_i \in \{0, 1, \ldots, n\}$. Sample $(j, p) \leftarrow \text{Mec}(b)$, and let $f = \max_i f_i$, $l = \min_{f_i = f} l_i$, $o_j = (j, p)$ and $o_i = \mathtt{sorry}$ for $i \neq j$.

possible abort: The execution is possibly *aborted*, as follows: Let j be the index of the winner. If $j < l$, then set $o_j \leftarrow \mathtt{sorry}$ with probability f, where P_j is the winner. If $j \geq l$, then set $o_j \leftarrow \mathtt{sorry}$ with probability $2f$.

offer contract, define outcome, side-channel: As in $\mathcal{C}^{\text{Mec}}_{\text{REJ}}$.

The recommend strategy $\pi^{\mathtt{unf}}_j$ for P_j is to input $b_j \leftarrow B_j(t_j)$ and $f_j = 0$ and $l_j = 1$, and give the local output $w_j = (t_j, o_j)$.

Figure 4: The Mediated Setting with Unfair Abort (and Reject) $(\mathcal{C}^{\mathtt{unf}}_{\text{Mec}}, \pi^{\mathtt{unf}}_B)$

6.1 Mediation with Unfair Abort

We consider a variant of the mediated setting with reject, where all parties have the possibility of having the mediation aborting, see Fig. 4.[3] The aborting is unfair in the sense that the party, P_k, inputting the largest probability has the advantage of setting a cut l dividing between different failure probabilities for winners P_j with $j < l$ and $j \geq l$. This introduces an advantage of suggesting the largest error probability. Nonetheless, if all parties, after seeing their type, still have a sufficiently large expected utility, it is an equilibrium to have all parties input $f_i = 0$. Intuitively, the only advantage of using $f_i > 0$ is the gain in privacy of not having the price leak if one looses. If P_k wins with probability w, this gain is at most $(1 - w)f\alpha$, where α is the weight of the information utility q_k. On the other side, when P_k would have won, an abort gives a lose of expected monetary utility. This is a cost of at least $w\frac{1}{2}f\gamma'$, where γ' is the expected utility given that P_j wins, and $\frac{1}{2}$ comes from the unfairness of the abort. If we let γ denote the expected monetary utility of P_j, then $\gamma = w\gamma'$. From *ex interim* individual rationality, $1 - w > 1$. To have an equilibrium, it is therefore enough that $\alpha \leq \frac{1}{2}\gamma$. The following theorem is proved in detail in Appendix D.

Theorem 2 *Let $\gamma \geq 0$ be the* ex interim *rationality of $(\pi^{rej}, \mathcal{C}_{\text{REJ}}, T)$ and let α be the weight of q. If π^{rej} is an ϵ-Nash equilibrium for $(\mathcal{C}_{\text{REJ}}, u, T)$ and $\alpha \leq \frac{1}{2}\gamma$, then π^{unf} is an ϵ-Nash equilibrium for $(\mathcal{C}^{unf}, u, T)$.*

Comments. Unfairness U in the abort, right now it is $U = 2$, would give the bound $\frac{1}{U}$. So, for U close to 1, α can be close to γ. The theorem can also be proved for collusions. In that case we need that any collusion always have a positive chance of winning even after seeing each others types. We omit the details, as this theorem is less interesting in our case—we do not know of a good auction mechanism for the mediated setting which tolerates collusion.

6.2 Assigning value to signed contracts

We want a protocol for the device $\mathcal{C}_{\texttt{int}}$. For this to be meaningful we need to make explicit how the Internet protocol allocates monetary utility. This is a fundamentally problematic issue as we are, after all, talking about a pure communication protocol which anyone can set up and run without no money being exchanged. As indicated in the introduction, we assign monetary value to a document if it is a possible winners outcome for Mec and is signed by all parties. In our setting we can make the reasonable assumption that the seller is willing to sell at any price (over some reservation price), and that he will sell to the first party presenting a properly signed document. This immediately assigns monetary value to properly signed contracts. A stronger assumption could be that we can use society to enforce signed contracts (cf. [14]).

In more detail, we assume that the key pair in $\mathcal{C}_{\texttt{int}}$ for each party P_i consists of a verification key pk_i for a digital signature scheme and the signing key sk_i. We call σ a *contract on* (j, p) if $\sigma = (\sigma_1, \ldots, \sigma_n)$ and σ_i is a valid signature on (j, p) under pk_i. We define $(o_1, \ldots, o_n) = \text{Out}(msg)$ by letting $o_i = (i, p)$ if P_i at some point sent $(i, p), \sigma$ to itself, where σ is a contract on (i, p). We let $o_j = \texttt{sorry}$ for all other parties. If no such contract was sent then $o_j = \texttt{sorry}$ for all parties. If several such contracts were sent, then the first one is used to define the outcome.[4]

[3]This has the potential to change the equilibria: If one is not the winner it is in ones advantage to have the execution abort, as it hides the result from the winner and therefore leaks less information on ones type.

[4]In each round messages are sent to one specific P_i, so it is well-defined which party sends a contract to itself first.

6.3 Mediation via a secure protocol

We proceed to implement the mediation with unfair abort (and reject) by a protocol $\pi^{\texttt{umed}}$. The idea is to compute the result as in the mediated setting, using a secure MPC protocol, but then release the result in a particular manner. The release phase will consist of E so-called epochs indexed $e = 1, \ldots, E$, each consisting of n tries indexed $i = 1, \ldots, n$. We index a try i within an epoch e by (e, i). In try (e, i) party P_i is given a value $V_{i,e}$, if the other parties allow it. The recommended strategy is to allow all deliveries, but as soon as a party has been denied a delivery, it will deny all parties their deliveries in all following tries. When P_j is the winner, then $V_{e,i} = \top$ for all $i \neq j$. For the winner P_j a random epoch $e_0 \in \{1, \ldots, E\}$ is chosen, and $V_{e_0,j} = \text{Contract}((j,p), sk)$ and $V_{e \neq e_0, j} = \top$. The epoch e_0 is chosen using a probabilistic function $e_0 \leftarrow \text{Epoch}(E)$, where $e_0 \in \{1, \ldots, E\}$ and $\Pr[e_0 = e] = 2^{-e}$ for $e = 1, \ldots, E-1$. When P_j receives $\text{Contract}((j,p), sk)$, it sends it to the seller (formally it sends it to itself and the device defines P_j to be the winner).

The recommend strategy $\pi_j^{\texttt{umed}}$ for P_j is as follows:

1. Receive (pk, sk_j) from the communication device.
2. In the rounds with point-to-point communication, run the code of P_j in a secure MPC for the following probabilistic function f:
   - Each P_i inputs some b_i and some (pk', sk_i). If all P_i input the same pk', and sk'_i is a signature key for pk'_i, then sample $(j,p) \leftarrow \text{Mec}(b)$, $e_0 \leftarrow \text{Epoch}(E)$, $\sigma = \text{Contract}((j,p), sk')$, $(V_{1,1}, \ldots, V_{E,n}) \leftarrow \text{Values}(\sigma, (e_0, j), E)$, $(S_{1,1}, \ldots, S_{E,n}) \leftarrow \text{Sharings}(V)$, and let $y_i = (S_{1,1}^{(i)}, \ldots, S_{E,n}^{(i)})$. Otherwise, let $y_i = \perp$. The output to P_i is y_i.

   Use inputs $b_j \leftarrow B_j(t_j)$ and $(pk', sk'_j) = (pk, sk_j)$.
3. Afterwards, initialize a variable $d_j \in \{\texttt{allegiance}, \texttt{defection}\}$, where $d_j = \texttt{defection}$ iff the secure MPC protocol outputs $y_j = \perp$. If $d_j \neq \texttt{defection}$, then parse y_j as shares $(S_{1,1}^{(j)}, \ldots, S_{E,n}^{(j)})$.
4. Use En rounds of point-to-point communication to sequentially run E *epochs*, each consisting of *tries* $i = 1, \ldots, n$. In each epoch e, try $i \neq j$ send $s_j = S_{e,i}^{(j)}$ to P_i if $d_j = \texttt{alligiance}$ and send $s_j = \perp$ to P_i otherwise. In epoch e, try j, compute $V_{e,j} = \oplus_{i=1}^{n} S_{e,j}^{(i)}$. If $V_{e,j} \neq \top$ and $V_{e,j}$ is not a valid contract, let $s_j = \perp$. Otherwise, let $s_j = \top$. If $s_j \neq \perp$ and $V_{e,j}$ is a valid contract for P_j, then input it to $\mathcal{C}^{\texttt{umed}}$ and send it to all other parties. If $s_j = \perp$, then let $d_j = \texttt{defection}$.
5. If a valid contract on (j,p) was received, give the local output $w_j = (t_j, (j,p))$. Otherwise, give the local output $w_j = (t_j, \texttt{sorry})$.

Figure 5: The Unmediated Protocol

We keep the index e_0 hidden from all parties. This ensures that denying another party a delivery will imply that one (with some positive probability) indirectly denied oneself the contract σ in a later try. This will ensure that denying another party a delivery has an expected negative utility for oneself. This holds for all epochs except the last one. Setting E large enough ensures that the probability 2^{-E} that $e_0 = E$ is negligible. Therefore denying in the last round gives at most a negligible advantage, and we have an ϵ-Nash.

It is convenient to have a notation for the $V_{e,i}$ values: For any bit-string σ, epoch index $e_0 \in \{1, \ldots, E\}$ and party index j, we define values $V = (V_{1,1}, \ldots, V_{1,n}, V_{2,1}, \ldots, V_{E,n}) = \text{Values}(\sigma, (e_0, j), E)$, where $V_{e_0,j} = \sigma$ and $V_{e,i} = \top$ for $e \neq e_0$ or $i \neq j$. We use a secure

MPC to compute sharings of the values $V_{e,i}$. Given inputs $(b_1, \ldots, b_n)$, the protocol securely samples $V = (V_{1,1}, \ldots, V_{1,n}, V_{2,1}, \ldots, V_{E,n})$ and generates sharings $(S_{1,1}, \ldots, S_{E,n}) \leftarrow \text{Sharings}(V)$, where $S_{e,i} = (S_{e,i}^{(1)}, \ldots, S_{e,i}^{(n)})$ is an n-out-of-n sharing of $V_{e,i}$. E.g., let all shares be uniformly random bit-strings, except that $V_{e,i} = \oplus_{j=1}^{n} S_{e,i}^{(j)}$. Then the protocol gives all $S_{e,i}^{(j)}$ to P_j. The MPC protocol is chosen to tolerate the active corruption of up to $t = n - 1$ parties. With this threshold termination cannot be guaranteed. The protocol should, however, guarantee that all parties P_j which received an output $y_j \neq \perp$, where $\perp$ is some designated error symbol, received a correct output. Furthermore, the protocol should guarantee that $y_j \neq \perp$ for all parties if all parties followed the protocol.

After the secure MPC protocol terminates, the parties reconstruct the sharings. Details are given in Fig. 5. We show that this protocol is an implementation of the mediated setting with unfair abort. In particular, in Appendix E we show the following theorem.

Theorem 3 *$(\pi^{umed}, \mathcal{C}^{umed})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\pi^{unf}, \mathcal{C}^{unf})$ for all α.*

The intuition is that not sending a share to some other party corresponds to aborting with some probability f_j. Making the MPC protocol fail corresponds to aborting with probability 1 ($f_i = \frac{1}{2}$, $l_i = 0$). Not sending the contract to oneself corresponds to rejecting an offered contract. Combining Theorems 1–3 we have that:

Corollary 1 *If π^{rej} is an ϵ-Nash equilibrium for $(\mathcal{C}^{\text{Mec}}_{\text{REJ}}, u, T)$, and $\alpha < \frac{1}{2}\gamma$, then π^{umed}_{Mec} is an ϵ-Nash equilibrium for $(\mathcal{C}^{umed}, u, T)$.*

This allows to take the mechanism from Section 4 and compile it into an ϵ-Nash protocol for an Internet-like setting. Note that Theorem 3 is much stronger than needed to facilitate this. We do, however, still find the extra strength of Theorem 3 assuring: Even though the protocol from Section 4 does not tolerate collusion, Theorem 3 at least guarantees that collusions in the protocol for the Internet have *no more* incentive to deviate than in the designed mechanism.

References

[1] I. Abraham, D. Dolev, R. Gonen, and J. Halpern. Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In *PODC '06: Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 53–62, New York, NY, USA, 2006. ACM.

[2] P. Bogetoft, I. Damgård, T. Jakobsen, K. Nielsen, J. Pagter, and T. Toft. A practical implementation of secure auctions based on multiparty integer computation. In *Proceedings of 10th International Conference on Financial Cryptography and Data Security*, volume 4107 of *Lecture Notes in Computer Science*, pages 142–147. Springer, 2006.

[3] P. G. Bradford, S. Park, and M. H. Rothkopf. Protocol completion incentive problems in cryptographic Vickrey auctions. Technical Report RRR 3-2004, Rutgers Center for Operations Research (RUTCOR), 2004.

[4] Y. Dodis, S. Halevi, and T. Rabin. A cryptographic solution to a game theoretic problem. In *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference*, volume 1880 of *Lecture Notes in Computer Science*, pages 112–130. Springer, August 2000.

[5] G. Escamocher, P. B. Miltersen, and R. Santillan-Rodriguez. Existence and computation of equilibria of first-price auctions with integral valuations and bids. Manuscript, 2008.

[6] S. D. Gordon and J. Katz. Rational secret sharing, revisited. In *Proceedings of 5th International Conference on Security and Cryptography for Networks*, pages 229–241, 2006.

[7] J. Halpern and R. Pass. Game theory with costly computation. Manuscript, 2008.

[8] J. Halpern and V. Teague. Rational secret sharing and multiparty computation: extended abstract. In *Proceedings of 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 623–632, 2004.

[9] S. Izmalkov, M. Lepinski, and S. Micali. Rational secure computation and ideal mechanism design. In *Proceedings of 46nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 585–594, 2005.

[10] G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In *Proceeding of Fifth Theory of Cryptography Conference (TCC)*, volume 4948 of *Lecture Notes in Computer Science*, pages 320–339. Springer, 2008.

[11] G. Kol and M. Naor. Games for exchanging information. To appear at the 40th Annual ACM Symposium on Theory of Computing (STOC), 2008.

[12] M. Lepinksi, S. Micali, and abhi shelat. Collusion-free protocols. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 543–552, New York, NY, USA, 2005. ACM.

[13] M. Lepinski, S. Micali, C. Peikert, and A. Shelat. Completely fair SFE and coalition-safe cheap talk. In *Proceedings of the twenty-third annual ACM symposium on Principles of Distributed Computing (PODC)*, pages 1–10, 2004.

[14] A. Y. Lindell. Legally-enforceable fairness in secure two-party computation. In T. Malkin, editor, *Proceedings of CT-RSA*, volume 4964 of *Lecture Notes in Computer Science*, pages 121–137. Springer, 2008.

[15] A. Lysyanskaya and N. Triandopoulos. Rationality and adversarial behavior in multi-party computation. In *Proceedings of Advances in Cryptology (CRYPTO)*, pages 180–197, August 2006.

[16] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *EC '99: Proceedings of the 1st ACM conference on Electronic commerce*, pages 129–139, New York, NY, USA, 1999. ACM.

[17] S. J. Ong, D. Parkes, A. Rosen, and S. Vadhan. Fairness with an honest minority and a rational majority. Manuscript, 2008.

[18] D. C. Parkes, M. O. Rabin, S. M. Shieber, and C. A. Thorpe. Practical secrecy-preserving, verifiably correct and trustworthy auctions. In *ICEC '06: Proceedings of the 8th international conference on Electronic commerce*, pages 70–81, New York, NY, USA, 2006. ACM.

[19] Y. Shoham and M. Tennenholtz. Non-cooperative computation: Boolean functions with correctness and exclusivity. *Theoretical Comput. Sci.*, 343(2):97–113, 2005.

Appendix

A The mediator as communication device

Mediator. We can model a mechanism as a function which given an input b_i from each P_i computes $(o_1, \ldots, o_n) \leftarrow \mathrm{Mec}(b_1, \ldots, b_n)$, where o_i is the outcome for P_i. By the revelation principle, we could consider only mechanisms where b_i is the type of P_i. For later use we will, however, consider mechanisms where each P_i submit some function $b_i = B_i(t_i)$ of its type t_i. The full *mechanism* is then specified by $(B_1, \ldots, B_n, \mathrm{Mec}, u)$. In the mediated setting, the communication device plays the role of a trusted mediator which computes Mec for the parties and sends back the outcomes. The details are given in Fig. 6.

The communication device $\mathcal{C}^{\mathtt{med}}_{\mathrm{Mec}}$ works as follows:

1. Each P_i inputs some b_i.
2. Sample $(o_1, \ldots, o_n) \leftarrow \mathrm{Mec}(b)$, where $b = (b_1, \ldots, b_n)$.
3. Output o_i to P_i.

The recommend strategy $\pi^{\mathtt{med}}_j$ for each P_j is to input $b_j \leftarrow B_j(t_j)$, and output $w_j = (t_j, o_j)$.

Figure 6: The Mediated Setting $(\mathcal{C}^{\mathtt{med}}_{\mathrm{Mec}}, \pi^{\mathtt{med}}_B)$ for mechanism $(B_1, \ldots, B_n, \mathrm{Mec})$

The utility of P_i is then given as $u_i(t, o, w)$. In the typical mediated setting the utility only depends on types and outcomes, i.e., $u_i(t, o, w) = u_i(t, o)$. As an example, we can model a Vickrey auction by $B_i(t_i) = t_i$ and letting $o_i = (i, p)$ for the P_i giving the highest bid b_i (draws are broken by randomization), and p being the value of the second highest bid, and letting $o_j = \mathtt{sorry}$ for all other parties but the winner. The utility is $u_j(t, o) = 0$ for all P_j with $o_j = \mathtt{sorry}$ and $u_i(t, o) = t_i - p$ when $o_i = (i, p)$.

B Proof of Lemma 1

Proof: We show a slightly stronger result. Consider the following class of *predictable* one-winner mechanisms satisfying the following property: when a bidder bids, he does not know if he will win or not; but if he does win, the output from the mediator depends only on his *own* input to the mechanism. Clearly the first-price auction is an example as the winner gave the maximal bid and receives (i, p), where p is the maximal bid.

For any predictable one-winner mechanism, any equilibrium in the mediated setting without reject, where the recommended strategy does not use the reject option, induces an equilibrium in the mediated setting with reject. The proof is as follows. First, the equilibrium in the mediated setting without reject certainly induces a strategy profile in the setting with reject. Now consider a deviation $\pi^{\mathtt{rej}*}_i$ in the setting with reject. We can run this strategy in the mediated setting as a strategy $\pi^{\mathtt{med}*}_i$. The strategy $\pi^{\mathtt{med}*}_i$ runs $\pi^{\mathtt{rej}*}_i$ until it gives a bid b'_i. Then it returns to $\pi^{\mathtt{rej}*}_i$ the outcome O_i that $\mathcal{C}_{\mathrm{REJ}}$ would offer to $\pi^{\mathtt{rej}*}_i$ if player i is the winner (it can by assumption predict this value).

- If this makes $\pi^{\mathtt{rej}*}_i$ try to reject the contract, then $\pi^{\mathtt{med}*}_i$ inputs $b_i = B_i(t_i)$ to $\mathcal{C}^{\mathtt{med}}$ (i.e., the bid recommended by Mec).
- If it makes $\pi^{\mathtt{rej}*}_i$ accept the contract, then $\pi^{\mathtt{med}*}_i$ inputs b'_i to $\mathcal{C}^{\mathtt{med}}$.

Then $\pi_i^{\mathtt{med}*}$ receives a result from $\mathcal{C}^{\mathtt{med}}$.

- If it is $o_i = \mathtt{sorry!}$, then it rewinds $\pi_i^{\mathtt{rej}*}$ to the point where it output b_i' and reruns it from this point, now giving it the result $o_i = \mathtt{sorry!}$ as if coming from $\mathcal{C}_{\mathrm{REJ}}$. When $\pi_i^{\mathtt{rej}*}$ produces a local output w_i, $\pi_i^{\mathtt{med}*}$ gives the local output w_i.
- If $o_i \neq \mathtt{sorry!}$, then $o_i = O_i$, where O_i is the value predicted above. In this case $\pi_i^{\mathtt{med}*}$ simply runs $\pi_i^{\mathtt{rej}*}$ until it produces a local output w_i, and $\pi_i^{\mathtt{med}*}$ gives the local output w_i.

We do a proof along the lines of Theorem 2. Let W be the event that party i wins and let A be the event that $\pi_i^{\mathtt{rej}*}$ accepts an offered contract. We look at three cases.

- If A, then the monetary outcome for party i is clearly the same in the two setting, as b_i' is played in both setting and the contract accepted in the setting with reject. In addition, the value w_i output by party i is by construction computed from the same view in both settings (as is w_j for all other P_j). Therefore also the privacy utilities are identical in the two settings.
- If W and $\bar{A}$, then the monetary utility in the mediated setting is at least μ', where μ' is the expected monetary utility for party i playing $\pi_i^{\mathtt{med}}$ *conditioned* on W and $\bar{A}$. In the setting with reject the monetary utility is 0 because $\bar{A}$. Since the difference between the privacy utilities in the two settings is at most α, it follows that $\pi_i^{\mathtt{med}*}$ gets an extra utility over $\pi_i^{\mathtt{rej}*}$ of at least $\mu' - \alpha$.
- If $\bar{W}$ and $\bar{A}$, then the monetary utility is 0 in both cases. The difference in the privacy utility is at most α. It follows that $\pi_i^{\mathtt{rej}*}$ gets an extra utility over $\pi_i^{\mathtt{med}*}$ of at most α.

It follows that $\pi_i^{\mathtt{med}*}$ gets an extra utility over $\pi_i^{\mathtt{rej}*}$ of at least

$$\begin{aligned}\Pr[A]0 + \Pr[W,\bar{A}](\mu' - \alpha) - \Pr[\bar{W},\bar{A}]\alpha &= \Pr[W,\bar{A}]\mu' - \alpha(\Pr[W,\bar{A}] + \Pr[\bar{W},\bar{A}]) \\ &= \Pr[W,\bar{A}]\mu' - \alpha\Pr[\bar{A}]\ .\end{aligned}$$

Let μ'' denote the expected monetary utility of party i playing the recommended strategy conditioned on $\bar{W}$ and $\bar{A}$. Clearly, $\mu'' = 0$ (as $\bar{W}$). Let μ denote the expected monetary utility of party i playing the recommended strategy conditioned on $\bar{A}$. Since the decision to reject or not can be taken given the type, we have that $\Pr[\bar{A}]\mu \geq \Pr[\bar{A}]\gamma$. We get that

$$\begin{aligned}\Pr[W,\bar{A}]\mu' - \alpha\Pr[\bar{A}] &= \Pr[W,\bar{A}]\mu' + \Pr[\bar{W},\bar{A}]\mu'' - \alpha\Pr[\bar{A}] \\ &= \Pr[\bar{A}]\mu - \alpha\Pr[\bar{A}] \\ &\geq \Pr[\bar{A}]\gamma - \alpha\Pr[\bar{A}] \\ &= \Pr[\bar{A}](\gamma - \alpha) \geq 0\ .\end{aligned}$$

Intuitively, this bound just says that where the strategy would reject, it would have been off playing as recommended and then accepting. Since $\pi_i^{\mathtt{rej}*}$ gets less utility by deviating than does $\pi_i^{\mathtt{med}*}$ and $\pi_i^{\mathtt{med}*}$ is deviating in an (ϵ-)Nash equilibrium, it follows that $\pi_i^{\mathtt{rej}*}$ gets no (at most negligible) utility out of deviating. □

C Proof of Theorem 1

Proof: That the notion is transitive is trivial.

To show that the notion is t-resilient ϵ-Nash preserving, assume that $(\gamma, \mathcal{D}, T)$ is a t-resilient ϵ-Nash. We show that so is $(\pi, \mathcal{C}, T)$. Let $C \subset \{1, \ldots, n\}$, $|C| \leq t$, let $l \in C$, and let π^*_C be a strategy for P_C for $\mathcal{C}$. By $(\pi, \mathcal{C})$ giving no more incentive to deviate than $(\gamma, \mathcal{D})$ there exists γ^*_C such that

$$u_l((\pi^*_C, \pi_{-C}), \mathcal{C}, T) \leq u_l((\gamma^*_C, \gamma_{-C}), \mathcal{D}, T) + \epsilon_1 .$$

By $(\gamma, \mathcal{D})$ being a t-resilient ϵ-Nash, we have that

$$u_l((\gamma^*_C, \gamma_{-C}), \mathcal{D}, T) \leq u_l(\gamma, \mathcal{D}, T) + \epsilon_2 .$$

By $(\pi, \mathcal{C})$ having no less utility we have that

$$u_l(\gamma, \mathcal{D}, T) \leq u_l(\pi, \mathcal{C}, T) + \epsilon_3 .$$

It follows that

$$u_l((\pi^*_C, \pi_{-C}), \mathcal{C}, T) \leq u_l(\pi, \mathcal{C}, T) + \epsilon ,$$

where $\epsilon = \epsilon_1 + \epsilon_2 + \epsilon_3$ is negligible.

To show that the utility profile is negligibly close to that of $(\gamma, \mathcal{D})$ it is sufficient to prove that there exists a negligible ϵ such that $u_l(\pi, \mathcal{C}, T) \geq u_l(\gamma, \mathcal{D}, T) - \epsilon$ and $u_l(\pi, \mathcal{C}, T) \leq u_l(\gamma, \mathcal{D}, T) + \epsilon$ for all P_l. From no less utility it follows that $u_l(\pi, \mathcal{C}, T) \geq u_l(\gamma, \mathcal{D}, T) - \epsilon_1$. From no more incentive to deviate with $C = \emptyset$ it follows that $u_l(\pi, \mathcal{C}, T) \leq u_l(\gamma, \mathcal{D}, T) + \epsilon_2$. Let $\epsilon = \epsilon_1 + \epsilon_2$. □

D Proof of Theorem 2

Proof: For sake of proof we can change to output code **possible abort** code in Fig. 4 to work as follows: Sample a bit a with $\Pr[a = 0] = 1 - f$ and sample a bit b with $\Pr[b = 0] = (1 - f)^{-1}(1 - 2f)$. Now set $o_j \leftarrow \texttt{sorry}$ if either ($j < l$ and $a = 1$) or ($j \geq l$ and ($a = 1$ or $b = 1$)). Note that the probabilities of aborting were not changed, as the probability that $a = 1$ is f and the probability that $a = 0$ and $b = 0$ is $1 - 2f$, and thus the probability that $a = 1$ or $b = 1$ is $2f$. In the below proof it is, however, convenient to have the independent random bits a and b defined.

Consider any deviating strategy $\pi^{\texttt{unf}^*}_j$ for $(\mathcal{C}^{\texttt{unf}}, u, T)$. For ease of presentation only, assume that f_j and b_j are deterministic functions of t_j. In addition, we consider only the case where the deviator P_j uses $l_j = j + 1$; If $j = n$, then $l_j = 1$. Using a larger l_j will only give some of the other parties a higher probability of getting the result, while keeping the probability that P_j gets the result the same. The techniques needed for handling this case is an easy special case of the techniques used below.

Consider the following deviating strategy $\pi^{\texttt{rej}^*}_j$ for the mediated setting with reject: It runs $\pi^{\texttt{unf}^*}_j(t_j)$ to get (b_j, l_j, f_j). Then it samples a bit a, where $a = 0$ with probability $1 - f_j$ and flips a uniformly random bit b, where $b = 0$ with probability $(1 - f_j)^{-1}(1 - 2f_j)$.

- If $a = 0$ then $\pi^{\texttt{rej}^*}_j$ inputs b_j and accepts the offered contract if $\pi^{\texttt{unf}^*}_j(t_j)$ would have done so.
- If $a = 1$, then it inputs $B_j(t_j)$ and always accepts the contract if it is offered by the device.

The idea is to play like $\pi^{\texttt{unf}^*}_j(t_j)$ when there is no abort and to play the recommended strategy when $\pi^{\texttt{unf}^*}_j(t_j)$ would have aborted. The intuition is that playing as recommended is better than aborting.

In the recommended strategies $\pi^{\texttt{rej}}_j$ and $\pi^{\texttt{unf}}_j$, and any other setting where t_j is known, we can also compute (b_j, l_j, f_j) from t_j as above and flip bits a and b as above, if they are not already defined by the device. In all such settings, let A be the event that $a = 1$ and let B be the event that $b = 1$.

We use $u_j^{\dots,\neg A}$ to denote the expected utility given that A does not occur. We start by bounding $p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{\mathtt{unf},\neg A})$.

It is clear that conditioned on A occurring, $u_j^{*,\mathtt{rej}} = u_j^{\mathtt{rej}}$, as both $\pi_j^{\mathtt{rej}*}$ and $\pi_j^{\mathtt{rej}}$ input $B_j(t_j)$ and accept the offered contract when $a = 1$. We use $u_j^{\dots,A}$ to denote the expected utility given the event A, and thus write

$$u_j^{*,\mathtt{rej},A} = u_j^{\mathtt{rej},A} \; .$$

It then follows from $\pi^{\mathtt{rej}}$ being an ϵ-Nash that

$$\begin{aligned} u_j^{*,\mathtt{rej}} - u_j^{\mathtt{rej}} &= p(A)(u_j^{*,\mathtt{rej},A} - u_j^{\mathtt{rej},A}) + p(\neg A)(u_j^{*,\mathtt{rej},\neg A} - u_j^{\mathtt{rej},\neg A}) \\ &= p(\neg A)(u_j^{*,\mathtt{rej},\neg A} - u_j^{\mathtt{rej},\neg A}) \le \epsilon \; . \end{aligned}$$

It is clear that $u_j^{\mathtt{rej},\neg A} = u_j^{\mathtt{unf},\neg A}$ (we are looking at the recommended strategies), which then implies that

$$p(\neg A)(u_j^{*,\mathtt{rej},\neg A} - u_j^{\mathtt{unf},\neg A}) \le \epsilon \; .$$

We get that

$$\begin{aligned} p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{\mathtt{unf},\neg A}) &= p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{*,\mathtt{rej},\neg A}) + p(\neg A)(u_j^{*,\mathtt{rej},\neg A} - u_j^{\mathtt{unf},\neg A}) \\ &\le p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{*,\mathtt{rej},\neg A}) + \epsilon \; . \end{aligned}$$

Let L denote the event that $j = l$, i.e., that P_l is the winner. We have that

$$\begin{aligned} &p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{*,\mathtt{rej},\neg A}) = \\ &p(\neg A, L)(u_j^{*,\mathtt{unf},\neg A,L} - u_j^{*,\mathtt{rej},\neg A,L}) + p(\neg A, \neg L)(u_j^{*,\mathtt{unf},\neg A,\neg L} - u_j^{*,\mathtt{rej},\neg A,\neg L}) \end{aligned}$$

and

$$u_j^{*,\mathtt{unf},\neg A,L} = u_j^{*,\mathtt{rej},\neg A,L} \; ,$$

which implies that

$$\begin{aligned} p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{*,\mathtt{rej},\neg A}) =& p(\neg A, \neg L)(u_j^{*,\mathtt{unf},\neg A,\neg L} - u_j^{*,\mathtt{rej},\neg A,\neg L}) \\ =& p(\neg A, \neg L, B)(u_j^{*,\mathtt{unf},\neg A,\neg L,B} - u_j^{*,\mathtt{rej},\neg A,\neg L,B}) + \\ & p(\neg A, \neg L, \neg B)(u_j^{*,\mathtt{unf},\neg A,\neg L,\neg B} - u_j^{*,\mathtt{rej},\neg A,\neg L,\neg B}) \; . \end{aligned}$$

Since $u_j^{*,\mathtt{unf},\neg A,\neg L,\neg B} = u_j^{*,\mathtt{rej},\neg A,\neg L,\neg B}$, it follows that

$$\begin{aligned} p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{*,\mathtt{rej},\neg A}) &= p(\neg A, \neg L, B)(u_j^{*,\mathtt{unf},\neg A,\neg L,B} - u_j^{*,\mathtt{rej},\neg A,\neg L,B}) \\ &= p(\neg A, \neg L, B)(q_j^{*,\mathtt{unf},\neg A,\neg L,B} - q_j^{*,\mathtt{rej},\neg A,\neg L,B}) \\ &\le p(\neg A, \neg L, B)\alpha \; , \end{aligned}$$

where α is the weight of q. So,

$$p(\neg A)(u_j^{*,\mathtt{unf},\neg A} - u_j^{\mathtt{unf},\neg A}) \le p(\neg A, \neg L, B)\alpha + \epsilon \; . \tag{1}$$

We now bound $p(A)(u_j^{*,\mathtt{unf},A} - u_j^{\mathtt{unf},A})$. We have that

$$\begin{aligned} p(A, L)(u_j^{*,\mathtt{unf},A,L} - u_j^{\mathtt{unf},A,L}) &= p(A, L)(q_j^{*,\mathtt{unf},A,L} - r_j^{\mathtt{unf},A,L} - q_j^{\mathtt{unf},A,L}) \\ &\le -p(A, L)r_j^{\mathtt{unf},A,L} + \epsilon' \; , \end{aligned}$$

as $q_j^{*,\mathtt{unf},A,L} \leq q_j^{\mathtt{unf},A,L} + \epsilon'$, by q_j preferring collection of knowledge. Since $r_j^{\mathtt{unf},A,\neg L} = 0$, we have that

$$\begin{aligned} p(A,L) r_j^{\mathtt{unf},A,L} &= p(A,L) r_j^{\mathtt{unf},A,L} + p(A,\neg L) r_j^{\mathtt{unf},A,\neg L} \\ &= p(A) r_j^{\mathtt{unf},A} \geq p(A)\gamma \ , \end{aligned}$$

as A depends only on t_j, which means that there is individual rationality γ given A. Furthermore,

$$p(A,\neg L)(u_j^{*,\mathtt{unf},A,\neg L} - u_j^{\mathtt{unf},A,\neg L}) = p(A,\neg L)(q_j^{*,\mathtt{unf},A,\neg L} - q_j^{\mathtt{unf},A,\neg L}) \leq p(A,\neg L)\alpha \ .$$

So, $p(A)(u_j^{*,\mathtt{unf},A} - u_j^{\mathtt{unf},A}) \leq -p(A,L) r_j^{\mathtt{unf},A,L} + \epsilon' + p(A,\neg L)\alpha$, which with (1) implies that

$$u_j^{*,\mathtt{unf},A} - u_j^{\mathtt{unf},A} \leq (p(\neg A,\neg L,B) + p(A,\neg L))\alpha - \gamma p(A) + \epsilon + \epsilon' \ ,$$

where $\epsilon + \epsilon'$ is negligible. So, it is sufficient that $\gamma p(A) \geq (p(\neg A,\neg L,B) + p(A,\neg L))\alpha$. We have chosen B such that the probability that B occurs and A does not occur is equal to the probability that A occurs. So, $p(\neg A,\neg L,B) = p(A,\neg L)$ and $(p(\neg A,\neg L,B) + p(A,\neg L))\alpha = p(A,\neg L)\alpha$. So, it is sufficient that $\alpha \leq \frac{1}{2}\gamma$. □

E Proof of Theorem 3

In this section we prove Theorem 3. For convenience of proof, we assume that the digital signature schemes used by the parties have unique signatures. This implies that given a contract σ and $n-1$ signing keys sk_i it is infeasible to compute a new contract $\sigma' \neq \sigma$ even on the same value (j,p), which allows a simplified proof.

We in addition need that

- $P 2^{-E+1}$ is negligible, where P is the largest bid which the system accepts; this can be accomplished by setting E appropriately. And,
- the probability of breaking the signature scheme σ using a poly-time algorithm should have the property that $\sigma(\kappa)P$ is negligible; this can be accomplished by e.g. letting P be polynomial in κ and using a signature scheme secure against poly-time adversaries.

The security needed for the multiparty computation protocol is specified below, when we need it.

E.1 Fleshing out

Below we will a number of times prove that some $(\pi^{\mathtt{hyb}(x+1)}, \mathcal{C}^{\mathtt{hyb}(x+1)})$ is a $n-1$-resilient ϵ-implementation of $(\pi^{\mathtt{hyb}x}, \mathcal{C}^{\mathtt{hyb}x})$. Almost all proofs will be based on proving three properties, and it is convenient to once and for all prove that these three properties are sufficient.

We will always prove that:

Q1: The distributions of $(t, o^{\mathtt{hyb}(x+1)}, w^{\mathtt{hyb}(x+1)})$ and $(t, o^{\mathtt{hyb}x}, w^{\mathtt{hyb}x})$ are the same, where $o^{\mathtt{hyb}(x+1)}$, $w^{\mathtt{hyb}(x+1)}$ are the outcome and local outputs in a random run of $(\pi^{\mathtt{hyb}(x+1)}, \mathcal{C}^{\mathtt{hyb}(x+1)})(t)$ and $o^{\mathtt{hyb}x}$, $w^{\mathtt{hyb}x}$ are the outcome and local outputs in a random run of $(\pi^{\mathtt{hyb}x}, \mathcal{C}^{\mathtt{hyb}x})(t)$.

This implies no less utility, as r_l and q_l are functions of (t,o,w,T) which has the same distribution in the two settings given Q1.

We then prove that for all $C \subset \{1,\dots,n\}$, $|C| \leq n-1$ and all deviations $\pi_C^{\mathtt{hyb}(x+1)^*}$, there exists a deviation $\pi_C^{\mathtt{hyb}x^*}$ and a negligible ϵ such that it holds for all P_l, $l \in C$ that:

Q2: $r_l((\pi_C^{\mathtt{hyb}x*}, \pi_{-C}^{\mathtt{hyb}x}), \mathcal{C}^{\mathtt{hyb}x}) \geq r_l((\pi_C^{\mathtt{hyb}(x+1)^*}, \pi_{-C}^{\mathtt{hyb}(x+1)}), \mathcal{C}^{\mathtt{hyb}(x+1)}) - \epsilon$.

Q3: $(t, \mathrm{view}_{(\pi_C^{\mathtt{hyb}(x+1)^*}, \pi_{-C}^{\mathtt{hyb}(x+1)}), \mathcal{C}^{\mathtt{hyb}(x+1)}}(t))$ and $(t, \mathrm{view}_{(\pi_C^{\mathtt{hyb}x*}, \pi^{\mathtt{hyb}x}{}_{-C}), \mathcal{C}^{\mathtt{hyb}x}}(t))$ are computationally indistinguishable, where $t \leftarrow T$ and $\mathrm{view}_{(\pi_C^*, \pi_{-C}), \mathcal{C}}(t) = \{w_i\}_{i=1}^n$ is the local outputs after a random run on t.

Properties Q2 and Q3 imply no more incentive to deviate: From Q2 we have that

$$\begin{aligned} u_l^{\mathtt{hyb}x}(t,o,w) &= r_l^{\mathtt{hyb}x}(t,o,w) + q_l^{\mathtt{hyb}x}(t,w) \\ &\geq (r_l^{\mathtt{hyb}(x+1)}(t,o,w) - \epsilon) + q_l^{\mathtt{hyb}x}(t,w) \\ &= u_l^{\mathtt{hyb}(x+1)}(t,o,w) - \epsilon + (q_l^{\mathtt{hyb}x}(t,w) - q_l^{\mathtt{hyb}(x+1)}(t,w)) \ . \end{aligned}$$

So, it is sufficient that $|q_l^{\mathtt{hyb}x}(t,w) - q_l^{\mathtt{hyb}(x+1)}(t,w)|$ is negligible, which follows from Q3 as q_l is a function of $(t, \mathrm{view}_{(\pi_C^*, \pi_{-C}), \mathcal{C}}(t))$ and assigns negligibly different valuation to computationally indistinguishable distributions.

In addition, when proving that $(\pi^{\mathtt{hyb}(x+1)}, \mathcal{C}^{\mathtt{hyb}(x+1)})$ is an $(n-1)$-resilient ϵNash implementation of $(\pi^{\mathtt{hyb}x}, \mathcal{C}^{\mathtt{hyb}x})$ we sometimes give a set B of strategies for $(\pi^{\mathtt{hyb}(x+1)}, \mathcal{C}^{\mathtt{hyb}(x+1)})$. We then show that for all $\pi_C^{\mathtt{hyb}(x+1)^*} \notin B$ that there exists a better strategy $\pi_C^{\mathtt{hyb}(x+1)^*} \in B$ in which all colluders do better, *a la* Q2 and Q3, and for all $\pi_C^{\mathtt{hyb}(x+1)^*} \in B$ in we show that there exists $\pi_C^{\mathtt{hyb}x*}$ meeting Q2 and Q3. This gives an implicit strategy mapping of all $\pi_C^{\mathtt{hyb}(x+1)^*}$ to some $\pi_C^{\mathtt{hyb}x*}$ meeting Q2 and Q3.

E.2 Hybrid 0

The details of Hybrid 0 are given in Fig. 7.

The communication device $\mathcal{C}^{\mathrm{h}0}$ works as follows:

compute result: Each P_i inputs some b_i along with some $(e_i, l_i) \in \{1, \ldots, E\} \times \{1, \ldots, n\} \cup \{(E+1, 1)\}$. Let $(e, l) = \min_i(e_i, l_i)$ [a] and sample $(j, p) \leftarrow \mathrm{Mec}(b)$ and $e_0 \leftarrow \mathrm{Epoch}(E)$. Let $o_j = (j, p)$ and $o_{i \neq j} = \mathtt{sorry}$.

possible abort: If $(e_0, j) \geq (e, l)$, then set $o_j = \mathtt{sorry}$.

offer contract, define outcome, side-channel: As in $\mathcal{C}_{\mathrm{REJ}}$.

The recommend strategy $\pi_j^{\mathrm{h}0}$ for P_j is as follows:

1. Input $b_j = t_j$ and $(e_j, l_j) = (E+1, 1)$.
2. If offered the contract, accept.
3. Output $w_j = (t_j, o_j)$.

[a] The minimum is taken according to the lexicographic ordering of (e, l).

Figure 7: Hybrid 0

Theorem 4 *$(\mathcal{C}^{h0}, \pi^{h0})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{unf}, \pi^{unf})$.*

Proof: Property Q1 is obvious.

Here we prove no more incentive to deviate directly (i.e., not through Q2 and Q3). For a deviating strategy $\pi_j^{\mathtt{h0}*}$ we let $\pi_j^{\mathtt{unf}*}(t_j)$ run $\pi_j^{\mathtt{h0}*}(t_j)$ until it outputs $(b_j, (e_j, l_j))$. Then $\pi_j^{\mathtt{unf}*}(t_j)$ inputs (b_j, l_j, f_j), where $f_j = 2^{-e_j}$, except that if $e_j = E + 1$, then it uses $f_j = 0$. It then runs $\pi_j^{\mathtt{h0}*}$ with $\mathcal{C}^{\mathtt{unf}}$. I.e., if offered a contract from $\mathcal{C}^{\mathtt{unf}}$, it inputs it to $\pi_j^{\mathtt{h0}*}$ and accepts if $\pi_j^{\mathtt{h0}*}$ accepts. It then inputs the final outcome o_i from $\mathcal{C}^{\mathtt{unf}}$ to $\pi_j^{\mathtt{h0}*}(t_j)$ and runs it until it outputs some w_j, which $\pi_j^{\mathtt{unf}*}(t_j)$ uses as local output. During this the colluding parties let the $\pi_j^{\mathtt{h0}*}(t_j)$ communicate using the side-channel as they desire, i.e., as they would have done in Hybrid 1.

In Hybrid 0, if $j < l$, then P_j is denied the result with probability 2^{-e}. If $j \geq l$, then P_j is denied the result with probability 2^{-e+1}. In Hybrid 1, if $j < l$, then P_j is denied the result iff $e_0 > e$. If $j \geq l$, then P_j is denied the result iff $e_0 \geq e$. If $e \leq E - 1$, then the probability that $e_0 > e$ is 2^{-e} and the probability that $e_0 \geq e$ is 2^{-e+1}, giving exactly the same probabilities. If $e = E + 1$, then the probability that $e_0 > e$ is 0 and the probability that $e_0 \geq e$ is 0. So, when $e_0 = E + 1$, the probabilities are the same in the two hybrids. If $e = E$, then the probability that $e_0 > e$ is 0 and the probability that $e_0 \geq e$ is 2^{-E+1}. So, when $e_0 > e$ the probability is 2^{-E} in one hybrid and 0 in the other. This shows that the statistical distance between (t, o, w, T) in the two settings is at most 2^{-E}. So, the statistical distance between $u_i(t, o, w, T)$ in the two settings are at most 2^{-E}. The difference between the expected value of $u_i(t, o, w, T)$ in the two hybrids is at most $P + \gamma$, where P is the polynomial upper bound on the price and γ is the weight of the privacy concern. By $\gamma < \alpha$ we have that $\gamma < P$. So, the difference is at most $2P2^{-E}$, which is negligible. □

E.3 Hybrid 1

The details of Hybrid 1 are given in Fig. 8.

The communication device $\mathcal{C}^{\mathtt{h1}}$ works as follows:

compute result: Each P_i inputs some b_i. Sample $(j, p) \leftarrow \text{Mec}(b)$, $e_0 \leftarrow \text{Epoch}(E)$, $(V_{1,1}, \ldots, V_{E,n}) \leftarrow \text{Values}((i, p), (e_0, j), E)$. Let $o_j = (j, p)$ and $o_{i \neq j} = \mathtt{sorry}$.

possible abort: Run sequential *epochs* $e = 1, \ldots, E$, each consisting of sequential *tries* $i = 1, \ldots, n$. In epoch e, try i, each $P_j \neq P_i$ inputs a bit $s_j \in \{\mathtt{send}, \mathtt{hold}\}$. If $s_j = \mathtt{send}$ for all P_j, then output $V_{i,e}$ to P_i. Otherwise, output $\perp$ to P_i. If the winner P_j does not receive (j, p) as output in some round, then set $o_j \leftarrow \mathtt{sorry}$.

offer contract, define outcome, side-channel: As $\mathcal{C}_{\text{REJ}}$.

The recommend strategy $\pi_j^{\mathtt{h1}}$ for P_j is as follows:

1. Input $b_j = t_j$.
2. Initialize a variable $d_j = \mathtt{allegiance}$.
3. In each epoch e, try $i \neq j$, input $s_j = \mathtt{send}$ iff $d_j = \mathtt{alligiance}$. In epoch e, try j, set $d_j = \mathtt{defection}$ if $\mathcal{C}^{\mathtt{h1}}$ outputs $\perp$.
4. If offered the contract, accept.
5. Output $w_j = (t_j, o_j)$.

Figure 8: Hybrid 1

Theorem 5 *$(\mathcal{C}^{h1}, \pi^{h1})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h0}, \pi^{h0})$.*

Proof: Property Q1 is obvious.

For Q2 and Q3 we specify strategy mappings. We let B be the set of strategies, where 1) no colluder inputs `hold` in a try (e,j) for $j \in C$ and 2) at most one colluder inputs `hold` in a try (e,j) for $j \notin C$, and does so at most once.

Given $\pi_C^{\mathtt{h1}*}$ which does not have property 1 we map it to one which does by having the colluders run $\pi_C^{\mathtt{h1}*}$, except that when instructed to input `hold` in try (j,p) for $j \in C$, a colluder inputs `send` instead and signals to P_j that it should ignore the output from the device. No matter the output, P_j then runs $\pi_C^{\mathtt{h1}*}$ as if the output from the device was $\perp$. In the end, each colluder gives the local output specified by $\pi_j^{\mathtt{h1}*}$. Clearly, all colluders end up giving the exact same local outputs. The same for non-colluders, as their outputs from the device were not affected by the transformation. The only possible difference is that some colluder P_j might get (j,p) instead of `sorry` in some try. If this happens it simply rejects the contract.

Given $\pi_C^{\mathtt{h1}*}$ which has property 1, but does not have property 2, we map it to one which has both by having the colluders run $\pi_C^{\mathtt{h1}*}$, except that when the first colluder inputs `hold` in try (e,j) for $j \notin C$, it signals to the other parties that it did so. In the following tries all colluders keep running $\pi_C^{\mathtt{h1}*}$, except that they actually input `send` in all tries. Since this will not affect the output of the device (as at least one honest party is inputting `hold` in each try already), the final local outputs of $\pi_C^{\mathtt{h1}*}$ will be the same. Clearly this also holds for the non-colluders, and clearly the outcome will be the same for all parties.

We then describe how to map a strategy $\pi_C^{\mathtt{h1}*}$ from B into one, $\pi_C^{\mathtt{h0}*}$, for Hybrid 0. In $\pi_C^{\mathtt{h0}*}$ some fixed colluder is elected coordinator, and all parties send their type to the coordinator. Then the coordinator runs $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$ as it would have been run in Hybrid 1, letting the colluders communicate via the side-channels until all $\pi_j^{\mathtt{h1}*}(t_j, \rho_C)$, $j \in C$ output some b_j. Here ρ_C denotes the randomness used by the colluders. The coordinator sends b_j to colluder P_j, which inputs $(b_j, (E+1, 1))$ to $\mathcal{C}^{\mathtt{h0}}$. Then the coordinator keeps running $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$ with a copy of $\mathcal{C}^{\mathtt{h1}}(\rho)$. It lets this copy run as if $V_{e,j} = \top$ for all e and all $j \in C$, and as if all non-colluders follow the recommended strategy. The coordinator runs the local $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$ until all colluders gave local outputs w_i. If, during this, some colluder in this copy of $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$ input `hold` to $\mathcal{C}^{\mathtt{h1}}$, the coordinator records the try (e,i) in which this happened. Otherwise, it sets $(e,i) = (E+1, 1)$. Then the coordinator P_j inputs $(b_j, (e,i))$ to $\mathcal{C}^{\mathtt{h0}}$. All colluders report their o_j to the coordinator. If no colluder received (j,p), then the coordinator sends w_i to each colluder P_i, which outputs w_i. Otherwise, if some colluder P_j received (j,p),[5] the coordinator samples a random e_0 such that (e_0, j) has the distribution that (e_0, j) has in Hybrid 1 given the fixed value of j and the condition that $(e_0, j) \leq (e,i)$. This can be done in expected poly-time by sampling $e_0 \leftarrow \mathrm{Epoch}(E)$ until $(e_0, j) \leq (e,i)$, but can also be done in strict poly-time, as required. Then the coordinator "rewinds" the local copy $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$ until try (e_0, j) and then runs $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$ forward from here, but this time letting $\mathcal{C}^{\mathtt{h1}}(\rho)$ return (j,p) to P_j in try (e_0, j). This makes the parties in the local copy $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$ eventually produce new local outputs w_i'. These the coordinator sends to the colluders, and colluder P_i outputs w_i'. If P_j accepts the contract in the local copy $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$, then the coordinator instructs P_j to accept the contract.

Imagine the copy of $\pi_C^{\mathtt{h1}*}(t_C, \rho_C)$, run by the coordinator, having been run in Hybrid 1 with $\mathcal{C}^{\mathtt{h1}}(\rho)$. Here we can define (e,i) as in Hybrid 0: The first try in which a colluder inputs `hold` if all colluders keep receiving $\top$ from the device. It is clear that P_j receives (j,p) if and only if $(e_0, j) \leq (e,i)$ in both hybrids, and that this happens with the same probability in both hybrids. Therefore the outcomes will be the same in both hybrids.

[5]When some P_j receivers the contract, it does not accept it yet. It waits for a go from the coordinator, as described below.

Furthermore, if $(e_0, j) > (e, i)$ or $j \notin C$, then in Hybrid 1 all colluders will receive $\top$ in all tries until the first colluder inputs `hold`, after which all colluders input `send` and receive `hold` in all tries, exactly as in the copy of $\pi_C^{\mathrm{h1}*}(t_C, \rho_C)$ run by the coordinator in Hybrid 0. Therefore both the outcomes and the local outputs will be exactly the same in this case.

If $(e_0, j) \leq (e, i)$ and $j \in C$, then in Hybrid 1 all colluders in $\pi_C^{\mathrm{h1}*}(t_C, \rho_C)$ will input `send` until the colluder P_j receives (j, p), as $(e_0, j) \leq (e, i)$. After this all colluders run as $\pi_C^{\mathrm{h1}*}(t_C, \rho_C)$ now runs after a colluder receives the result. This is exactly the same behaviour as the rewound and rerun copy of $\pi_C^{\mathrm{h1}*}(t_C, \rho_C)$ inside the coordinator in Hybrid 0. Therefore, also in this case, the outcome and the local outputs will be the same in both settings.

From the above it is easy to establish Q2 and Q3. □

E.4 Hybrid 2

By a communication device $\mathcal{C}$ handing out values $(v_1, \ldots, v_n)$ to parties $(P_1, \ldots, P_n)$ in *unfair round-robin* we mean the following. In round i, $\mathcal{C}$ tries to deliver v_i to P_i. In round i each party P_j inputs $s_j \in \{\texttt{send}, \texttt{hold}\}$. If $s_j = \texttt{send}$ for $j = 1, \ldots, n$, then $\mathcal{C}$ outputs v_i to P_i. Otherwise, $\mathcal{C}$ outputs some reserved failure symbol $\perp$ to P_i. The recommended strategy for P_j is always to input $d_j = \texttt{send}$ in rounds $i = 1, \ldots, j$ and to input $d_j = \texttt{send}$ in rounds $i = j+1, \ldots, n$ if and only if P_j did not receive output $\perp$ in round j.

In Hybrid 2 (Fig. 9) we introduce an initial unfair round-robin of some dummy values. If a party defects in this round, the other parties punish by defecting in all tries. This initial unfair round-robin will be expanded into the cryptographic secure sampling of sharings by later hybrids.

The communication device $\mathcal{C}^{\mathrm{h2}}$ works as follows:

compute result: Each P_i inputs some b_i. Sample $(j, p) \leftarrow \mathrm{Mec}(b)$, $e_0 \leftarrow \mathrm{Epoch}(E)$, $(V_{1,1}, \ldots, V_{E,n}) \leftarrow \mathrm{Values}((i, p), (e_0, j), E)$.

round-robin delivery: Deliver $(\top, \ldots, \top)$ to $(P_1, \ldots, P_n)$ in unfair round-robin.

possible abort: Run sequential epochs $e = 1, \ldots, E$, each consisting of sequential tries $i = 1, \ldots, n$. In epoch e, try i, each $P_j \neq P_i$ inputs a bit $s_j \in \{\texttt{send}, \texttt{hold}\}$. If $s_j = \texttt{send}$ for all P_j, then output $V_{i,e}$ to P_i. Otherwise, output $\perp$ to P_i. If the winner P_j does not receive (j, p) as output in some round, then set $o_j \leftarrow \texttt{sorry}$

offer contract, define outcome, side-channel: As $\mathcal{C}_{\mathrm{REJ}}$.

The recommend strategy π_j^{h2} for P_j is as follows:

1. Input $b_j = t_j$.
2. Use the standard recommended strategy for unfair round-robin, and afterwards initialize a variable $d_j \in \{\texttt{allegiance}, \texttt{defection}\}$, where $d_j = \texttt{defection}$ iff $\perp$ was received from the unfair round robin.
3. In each epoch e, try $i \neq j$ input $s_j = \texttt{send}$ iff $d_j = \texttt{alligiance}$. In epoch e, try j, set $d_j = \texttt{defection}$ if $\mathcal{C}^{\mathrm{h2}}$ outputs $\perp$.
4. If offered the contract, accept.
5. Output $w_j = (t_j, o_j)$.

Figure 9: Hybrid 2

Theorem 6 *$(\mathcal{C}^{h2}, \pi^{h2})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h1}, \pi^{h1})$.*

Proof: Q1 is trivial.

Strategy mapping. For properties Q2 and Q3 we must for each strategy $\pi_C^{\mathrm{h2}*}$ for $\mathcal{C}^{\mathrm{h2}}$ specify a strategy $\pi_C^{\mathrm{h1}*}$ for $\mathcal{C}^{\mathrm{h1}}$.

As for the proof of Theorem 5 we can assume that no colluder inputs `hold` in the initial unfair round-robin when it is the turn of another colluder. We can also assume that some fixed coordinator takes care of inputting `hold` in the turns of the non-colluders.

In $\pi_C^{\mathrm{h1}*}$ each $\pi_l^{\mathrm{h1}*}$ runs $\pi_l^{\mathrm{h2}*}$ as a sub-routine, and they let these $\pi_l^{\mathrm{h2}*}$ communicate using the side-channel, as they would have done in Hybrid 2.

First $\pi_l^{\mathrm{h1}*}$ inputs t_l to $\pi_l^{\mathrm{h2}*}$ to get b_l, which it inputs to $\mathcal{C}^{\mathrm{h1}}$. Then it internally runs the n rounds of the unfair round-robin with $\pi_l^{\mathrm{h2}*}$. Initially all colluders return $\top$ to $\pi_l^{\mathrm{h2}*}$ as if it came from $\mathcal{C}^{\mathrm{h2}}$. Let P_j denote the coordinator. If $\pi_j^{\mathrm{h2}*}$ inputs `hold` to $\mathcal{C}^{\mathrm{h2}}$ in some turn, then $\pi_j^{\mathrm{h1}*}$ informs the other colluders of that, and from that point on they all return $\perp$ to $\pi_l^{\mathrm{h2}*}$ as if it came from $\mathcal{C}^{\mathrm{h2}}$.

The En tries are run as follows:

If the coordinator input `send` in all rounds in the initial unfair round-robin, then $\pi_l^{\mathrm{h1}*}$ runs $\pi_l^{\mathrm{h2}*}$ with $\mathcal{C}^{\mathrm{h1}}$: It sends the same as $\pi_l^{\mathrm{h2}*}$ in all tries and returns the values from $\mathcal{C}^{\mathrm{h1}}$ to $\pi_l^{\mathrm{h2}*}$ as if they came from $\mathcal{C}^{\mathrm{h2}}$. In the end it adopts the local output of $\pi_l^{\mathrm{h2}*}$. This will produce the exact same distribution on outcomes and local outputs.

If the coordinator sent `hold` in some round in the initial unfair round-robin, then $\pi_l^{\mathrm{h1}*}$ runs $\pi_l^{\mathrm{h2}*}$ with $\mathcal{C}^{\mathrm{h1}}$, as follows: No matter what $\pi_l^{\mathrm{h2}*}$ sends, $\pi_l^{\mathrm{h1}*}$ sends `hold` in all tries, and then returns to $\pi_l^{\mathrm{h2}*}$ what it receives from $\mathcal{C}^{\mathrm{h1}}$. In the end it adopts the local output of $\pi_l^{\mathrm{h2}*}$. In Hybrid 2, the inputting of `hold` in the initial unfair round-robin would make the affected non-colluder input `hold` in all En tries, so in both hybrids $\mathcal{C}^{\mathrm{h1}}$ respectively $\mathcal{C}^{\mathrm{h2}}$ outputs $\perp$ to all colluders in all tries. Therefore the exact same distribution on local outputs is produces. It can also be seen that the outcomes are identically distributed in the two hybrids.

From the above it is easy to prove Q2 and Q3. □

E.5 Hybrid 3

In Hybrid 3 (Fig. 10) we essentially just introduce the distribution of random public keys and the outputting of a signature, see Fig. 10.

We show that introducing the distribution of the joint randomness vk did not disturb the equilibrium. The intuitive reason is that the recommended strategies do not use vk and therefore any P_l could in fact just have sampled its own vk.

Theorem 7 *$(\mathcal{C}^{h3}, \pi^{h3})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h2}, \pi^{h2})$.*

Proof: Q1 is trivial.

Strategy mapping. For properties Q2 and Q3 we must for each strategy $\pi_C^{\mathrm{h3}*}$ for $\mathcal{C}^{\mathrm{h3}}$ specify a strategy $\pi_C^{\mathrm{h2}*}$ for $\mathcal{C}^{\mathrm{h2}}$. For $l \in C$, the strategy $\pi_l^{\mathrm{h2}*}$ runs $\pi_l^{\mathrm{h3}*}$. First $\pi_l^{\mathrm{h2}*}$ inputs t_l to $\pi_l^{\mathrm{h3}*}$.

Then one of the $\pi_l^{\mathrm{h2}*}$ samples key pairs (vk_i, sk_i) for all P_i and sends them to the other parties P_C, and each P_j inputs (vk, sk_j) to $\pi_j^{\mathrm{h3}*}$. Then each P_l runs $\pi_l^{\mathrm{h3}*}$ internally. When it sends a message to another $\pi_j^{\mathrm{h3}*}$, then $\pi_l^{\mathrm{h2}*}$ sends the message to $\pi_j^{\mathrm{h2}*}$ which inputs it to $\pi_j^{\mathrm{h3}*}$. When $\pi_l^{\mathrm{h3}*}$ outputs some b_l, then it is input to $\mathcal{C}^{\mathrm{h2}}$. Then again it internally runs $\pi_l^{\mathrm{h3}*}$ as above. In addition it relays all outputs from $\pi_l^{\mathrm{h3}*}$ to $\mathcal{C}^{\mathrm{h2}}$ and all outputs from $\mathcal{C}^{\mathrm{h2}}$ to $\pi_l^{\mathrm{h3}*}$, except that if $\mathcal{C}^{\mathrm{h2}}$ outputs (l, p), then $\pi_l^{\mathrm{h2}*}$ inputs $\sigma = \mathrm{Contract}((l, p), sk)$, to $\pi_l^{\mathrm{h3}*}$. When $\pi_l^{\mathrm{h3}*}$ outputs some w_l, then $\pi_l^{\mathrm{h2}*}$ outputs w_l.

The communication device $\mathcal{C}^{\text{h}3}$ works as follows:

key distribution: For each P_i sample a key pair (vk_i, sk_i) and output (vk, sk_i) to P_i, where $vk = (vk_1, \ldots, vk_n)$.

compute result: Each P_i inputs some b_i. Sample $(j,p) \leftarrow \text{Mec}(b)$, $e_0 \leftarrow \text{Epoch}(E)$, $\sigma = \text{Contract}((j,p), sk)$, $(V_{1,1}, \ldots, V_{E,n}) \leftarrow \text{Values}(\sigma, (e_0, j), E)$.

round-robin delivery: Deliver $(\top, \ldots, \top)$ to $(P_1, \ldots, P_n)$ in unfair round-robin.

possible abort: Run sequential epochs $e = 1, \ldots, E$, each consisting of sequential tries $i = 1, \ldots, n$. In epoch e, try i, each $P_j \neq P_i$ inputs a bit $s_j \in \{\texttt{send}, \texttt{hold}\}$. If $s_j = \texttt{send}$ for all P_j, then output $V_{i,e}$ to P_i. Otherwise, output $\perp$ to P_i. If the winner P_j does not receive σ in some round, then set $o_j \leftarrow \texttt{sorry}$

offer contract, define outcome, side-channel: As $\mathcal{C}_{\text{REJ}}$.

The recommend strategy $\pi_j^{\text{h}3}$ is as $\pi_j^{\text{h}2}$.

Figure 10: Hybrid 3

Q3: Trivial, as the view seen by the copy of $\pi_C^{\text{h}3*}$ run by $\pi_C^{\text{h}2*}$ is identical to the view they have when run with $\mathcal{C}^{\text{h}3}$. The view is simply augmented with a random signature on the result.

Q2: Trivial using similar arguments. □

E.6 Hybrid 4

In Hybrid 4 (Fig. 11) we let the winner be the first one to input a valid signed contract. In Hybrid 3 the winner was the first to *receive* a valid contract and accept.

We show that changing the winner from being the first to receive a contract to the first to produce a contract does not disturb the ϵ-equilibrium. Intuitively, the reason is that inputting a contract before receiving one involves breaking the signature scheme of one of the other parties, which happens with negligible probability.

Theorem 8 *$(\mathcal{C}^{h4}, \pi^{h4})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h3}, \pi^{h3})$.*

Proof: Q1 holds for ϵ being the probability that a properly generated contract is rejected, which is 0 assuming that the signature schemes have perfect correctness.

Strategy mapping. For properties Q1 and Q2 we must for each P_l, $l \in C$ and strategy $\pi_l^{\text{h}4*}$ for $\mathcal{C}^{\text{h}4}$ specify a strategy $\pi_l^{\text{h}3*}$ for $\mathcal{C}^{\text{h}3}$. The strategy $\pi_l^{\text{h}3*}$ runs $\pi_l^{\text{h}4*}$. First $\pi_l^{\text{h}3*}$ inputs t_l to $\pi_l^{\text{h}4*}$ and then it simply runs $\pi_l^{\text{h}4*}$ with $\mathcal{C}^{\text{h}3}$. The strategies $\pi_l^{\text{h}3*}$ let their internal copies of $\pi_l^{\text{h}4*}$ communicate as they would in Hybrid 4.

Q2: The only way it can happen that $r_l(\pi_C^{\text{h}4*}, \pi_{-C}^{\text{h}4}) > r_l(\pi_C^{\text{h}3*}, \pi_{-C}^{\text{h}3})$ is that $\pi_l^{\text{h}4*}$ input a contract on (l, p') to $\mathcal{C}^{\text{h}3}$ respectively $\mathcal{C}^{\text{h}4}$ without having received a contract on (l, p'). With $\mathcal{C}^{\text{h}4}$ this can result in P_l winning, if no other parties were faster. With $\mathcal{C}^{\text{h}4}$ this does not result in P_l winning, as the winner is the one to *receive* a contract (and accept it). So, let q be the probability that some $\pi_l^{\text{h}4*}$ outputs a contract on (l, p') without having received a contract on (l, p'). Using a standard reduction to the unforgeability of any of the signature schemes of P_j, $j \notin C$ (recall that $|C| \leq n-1$) it can be shown that q is no larger than $\sigma(\kappa)$ — the security level of the signature scheme. When a contract

The communication device $\mathcal{C}^{\mathrm{h4}}$ works as follows:

key distribution: As `h3`.

compute result: Each P_i inputs some b_i. Sample $(j,p) \leftarrow \mathrm{Mec}(b)$, $e_0 \leftarrow \mathrm{Epoch}(E)$, $\sigma = \mathrm{Contract}((j,p), sk)$, $(V_{1,1}, \ldots, V_{E,n}) \leftarrow \mathrm{Values}(\sigma, (e_0, j), E)$.

round-robin delivery: As `h3`.

release contract: Run sequential epochs $e = 1, \ldots, E$, each consisting of sequential tries $i = 1, \ldots, n$. In epoch e, try i, each $P_j \neq P_i$ inputs a bit $s_j \in \{\texttt{send}, \texttt{hold}\}$. If $s_j = \texttt{send}$ for all P_j, then output $V_{i,e}$ to P_i. Otherwise, output $\perp$ to P_i.

accept contract: In all rounds allow all P_i to input some special value σ'. In the last round, output o_i to P_i, where the outcome o_i is computed as follows: If P_i was the first to input a contract σ' on some (i,p), then let $o_i = (i,p)$. Otherwise, let $o_i = \texttt{sorry}$.

side-channel: As $\mathcal{C}_{\text{REJ}}$.

The recommend strategy π_j^{h4} for P_j is as follows:

1. Input $b_j = t_j$.
2. Use the standard recommended strategy for unfair round-robin, and afterwards initialize a variable $d_j \in \{\texttt{allegiance}, \texttt{defection}\}$, where $d_j = \texttt{defection}$ iff $\perp$ was received from the unfair round robin.
3. In each epoch e, try $i \neq j$ input $s_j = \texttt{send}$ iff $d_j = \texttt{alligiance}$. In epoch e, try j, set $d_j = \texttt{defection}$ if $\mathcal{C}^{\mathrm{h4}}$ outputs $\perp$. If $\mathcal{C}^{\mathrm{h4}}$ outputs a contract σ, then input σ to $\mathcal{C}^{\mathrm{h4}}$.
4. If a valid contract on (j,p) was received, give the local output $w_j = (t_j, (j,p))$. Otherwise, give the local output $w_j = (t_j, \texttt{sorry})$.

Figure 11: Hybrid 4

on (l,p') is input to $\mathcal{C}^{\mathrm{h4}}$, the monetary utility is at most P — the upper bound on monetary utility. Therefore $r_l(\pi_C^{\mathrm{h4}*}, \pi_{-C}^{\mathrm{h4}}) - r_l(\pi_C^{\mathrm{h3}*}, \pi_{-C}^{\mathrm{h3}}) \leq \sigma(\kappa)P$. We have set P and $\sigma(\kappa)$ exactly such that $\sigma(\kappa)P$ is negligible.

Q3: This is almost trivial as the same strategies are run in both settings. The only complication is that if P_l, $l \in C$ inputs a valid contract before receiving one, the outcome o_j from the communication device $\mathcal{C}^{\mathrm{h3}}$ respectively $\mathcal{C}^{\mathrm{h4}}$ might differ. This, however, happens with negligible probability. □

E.7 Hybrid 5

In Hybrid 5 (Fig. 12) we hand out sharings of the values $V_{e,j}$ and we replace inputting `send` and `hold` with inputting the true share or not.

Theorem 9 *$(\mathcal{C}^{h5}, \pi^{h5})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h4}, \pi^{h4})$.*

Proof: Q1 is trivial.

Strategy mapping. For properties Q2 and Q3 we must for each P_l, $l \in C$ and each strategy $\pi_l^{\mathrm{h5}*}$ for $\mathcal{C}^{\mathrm{h5}}$ specify a strategy $\pi_l^{\mathrm{h4}*}$ for $\mathcal{C}^{\mathrm{h4}}$. The strategy $\pi_l^{\mathrm{h4}*}$ runs $\pi_l^{\mathrm{h5}*}$. First $\pi_l^{\mathrm{h4}*}$ inputs t_l and (vk, sk_l) to $\pi_l^{\mathrm{h5}*}$ and runs $\pi_l^{\mathrm{h5}*}$ to get b_l which it inputs to $\mathcal{C}^{\mathrm{h4}}$. During this, and below, the $\pi_l^{\mathrm{h5}*}$ are

The communication device $\mathcal{C}^{\mathrm{h5}}$ works as follows:

key distribution: As $\mathcal{C}^{\mathrm{h4}}$.

compute result: Each P_i inputs some b_i. Sample $(j,p) \leftarrow \mathrm{Mec}(b)$, $e_0 \leftarrow \mathrm{Epoch}(E)$, $\sigma = \mathrm{Contract}((j,p), sk)$, $(V_{1,1},\dots,V_{E,n}) \leftarrow \mathrm{Values}(\sigma,(e_0,j),E)$, $(S_{1,1},\dots,S_{E,n}) \leftarrow \mathrm{Sharings}(V)$. Let $y_i = (S_{1,1}^{(i)},\dots,S_{E,n}^{(i)})$, where $S_{e,j}^{(i)}$ is the share of P_i in $S_{e,j}$.

round-robin delivery: Deliver $(y_1,\dots,y_n)$ to $(P_1,\dots,P_n)$ in unfair round-robin.

release contract: Run sequential epochs $e = 1,\dots,E$, each consisting of sequential tries $i = 1,\dots,n$. In epoch e, try i, each $P_j \neq P_i$ inputs a value s_j. If $\oplus_j S_{e,i}^{(j)} = \oplus_j s_j$, then let $I_i = \top$. Otherwise, let $I_i = \bot$. Output I_i to P_i along with all the s_j.

accept contract, side-channel: As $\mathcal{C}^{\mathrm{h4}}$ respectively $\mathcal{C}_{\mathrm{REJ}}$.

The recommend strategy π_j^{h5} for P_j is as follows:

1. Input $b_j = t_j$.
2. Use the standard recommended strategy for unfair round-robin, and afterwards initialize a variable $d_j \in \{\texttt{allegiance}, \texttt{defection}\}$, where $d_j = \texttt{defection}$ iff $\bot$ was received from the unfair round robin.
3. In each epoch e, try $i \neq j$ input $s_j = S_{e,i}^{(j)}$ if $d_j = \texttt{alligiance}$ and input $s_j = \bot$ otherwise. In epoch e, try j, if $I_j \neq \bot$, then let $V_{e,j} = \oplus_{i=1}^{n} s_i$. If $V_{e,j}$ is a valid contract, then input it to $\mathcal{C}^{\mathrm{h5}}$. If $I_j = \bot$, then let $d_j = \texttt{defection}$.
4. If a valid contract on (j,p) was received, give the local output $w_j = (t_j,(j,p))$. Otherwise, give the local output $w_j = (t_j, \texttt{sorry})$.

Figure 12: Hybrid 5

allowed to communicate as in Hybrid 5. Then $\pi_l^{\mathrm{h4}*}$ runs $\pi_l^{\mathrm{h5}*}$ with $\mathcal{C}^{\mathrm{h4}}$ in the initial unfair round-robin. If the output is $\top$, then $\pi_l^{\mathrm{h4}*}$ lets y_l consist of random shares $S_{e,i}^{(l)}$ and inputs y_l to $\pi_l^{\mathrm{h5}*}$. If the output is $\bot$, then $\bot$ is input to $\pi_l^{\mathrm{h5}*}$. Then $\pi_l^{\mathrm{h4}*}$ runs $\pi_l^{\mathrm{h5}*}$ with $\mathcal{C}^{\mathrm{h4}}$ with the following four changes: When $\pi_l^{\mathrm{h5}*}$ outputs s_l (in epoch e, try i), it sends it to all colluders along with $S_{e,i}^{(l)}$. If $\oplus_j S_{e,i}^{(j)} = \oplus_j s_j$, then they all input $\texttt{send}$ to $\mathcal{C}^{\mathrm{h4}}$. Otherwise they all input $\texttt{hold}$. When $\mathcal{C}^{\mathrm{h4}}$ outputs $V_{e,l}$ (in epoch e, try l) which is $\top$ or a valid contract, then $\pi_l^{\mathrm{h4}*}$ inputs $(s_1,\dots,s_n)$ to $\pi_l^{\mathrm{h5}*}$, where s_j was sent to P_j for $j \in C$ and P_l picks s_i for $i \notin C$ to be uniformly random values such that $\oplus_{i=1}^{n} s_i = V_{e,l}$. When $\mathcal{C}^{\mathrm{h4}}$ outputs $V_{e,l} = \bot$, then some P_i input $\texttt{hold}$. For all P_i, $i \notin C$ doing so, use $s_i = \bot$. For the rest, pick s_i uniformly at random. When $\pi_l^{\mathrm{h5}*}$ produces a local output, $\pi_l^{\mathrm{h4}*}$ uses it as its own local output.

Q2: The two settings will deny the winner the value (j,p) with the exact same probabilities.

Q3: In both settings the shares seen by $\pi_l^{\mathrm{h5}*}$ are uniformly random sharings of the values $V_{e,i}$ when all shares are received. When less than n shares are received, the share are uniformly random and independent values in both settings (as at least one share is missing and any $n-1$ shares are uniformly random, independent values). Therefore the local outputs of $\pi_l^{\mathrm{h5}*}$ will have the same distribution in the two settings. □

E.8 Hybrid 6

In Hybrid 6 the indicator I_i of whether the reconstruction is correct is computed by the party P_i and not by the communication device. Since $\oplus_{i=1}^{n} s_i$ should be either $\top$ or a valid contract, P_i sets $I_i = \top$ iff this is the case.

The communication device $\mathcal{C}^{\mathrm{h6}}$ works as $\mathcal{C}^{\mathrm{h5}}$ with the following modification:

release contract: Run sequential epochs $e = 1, \ldots, E$, each consisting of sequential tries $i = 1, \ldots, n$. In epoch e, try i, each $P_j \neq P_i$ inputs a value s_j which is delivered to P_i.

The recommend strategy π_j^{h6} for P_j works as π_j^{h5} with the following modification:

3. In each epoch e, try $i \neq j$ input $s_j = S_{e,i}^{(j)}$ if $d_j = \texttt{alligiance}$ and input $s_j = \perp$ otherwise. In epoch e, try j, compute $V_{e,j} = \oplus_{i=1}^{n} s_i$. If $V_{e,j} \neq \top$ and $V_{e,j}$ is not a valid contract, let $I_j = \perp$. Otherwise, let $I_j = \top$. If $I_j \neq \perp$ and $V_{e,j}$ is a valid contract for P_j, then input the contract to $\mathcal{C}^{\mathrm{h6}}$. If $I_j = \perp$, then let $d_j = \texttt{defection}$.

Figure 13: Hybrid 6

We argue that this way of computing I_i produces the same result as when computed by the communication device, except with negligible probability.

Theorem 10 *$(\mathcal{C}^{h6}, \pi^{h6})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h5}, \pi^{h5})$.*

Proof:

Q1: Trivial as I_i has the same value, $\top$, no matter who computes it when all parties use the recommended strategies.

Strategy mapping. For properties Q2 and Q3 we must for each P_l, $l \in C$ and each strategy $\pi_l^{\mathrm{h6}*}$ for $\mathcal{C}^{\mathrm{h6}}$ specify a strategy $\pi_l^{\mathrm{h5}*}$ for $\mathcal{C}^{\mathrm{h5}}$. The strategy $\pi_l^{\mathrm{h5}*}$ runs $\pi_l^{\mathrm{h6}*}$ with $\mathcal{C}^{\mathrm{h5}}$, and the $\pi_l^{\mathrm{h6}*}$ are allowed to communicate as in Hybrid 5. The only modification needed is due to the fact that $\mathcal{C}^{\mathrm{h5}}$ outputs $I_l \in \{\perp, \top\}$. Since $\pi_l^{\mathrm{h6}*}$ does not expect to see such a value from, the strategy $\pi_l^{\mathrm{h5}*}$ does not relay this value to $\pi_l^{\mathrm{h6}*}$.

Properties Q2 and Q3 follow directly from the fact that the two different ways of computing I_i lead to the same result unless the security of the signature scheme was broken, which is assumed to happen with negligible probability. To prove this, it is sufficient to prove that if P_l, $l \in C$ send s_l such that $\oplus_{j \in C} s_j \neq \oplus_{j \in C} S_{e,i}^{(l)}$ for $i \notin C$, then $I_i = \perp$ except with negligible probability.

Note that if $(S_{e,i}^{(1)}, \ldots, S_{e,i}^{(n)})$ is a sharing of $V_{e,i}$ and the colluders P_l send s_j such that $\oplus_{j \in C} s_j \neq \oplus_{j \in C} S_{e,i}^{(l)}$, then P_i computes $V'_{e,i} = V_{e,i} \oplus \Delta_l$, where $\Delta_l = (\oplus_{j \in C} s_j) \oplus (\oplus_{j \in C} S_{e,i}^{(j)})$. And, Δ_l is a value which the colluders together could compute efficiently by pooling their views. Assume now that $\Delta_l \neq 0$ and $I_i = \top$ — the case we should argue happens with negligible probability.

We know that $V_{e,i} = \top$ or that $V_{e,i}$ is a valid contract. Assume first that $V_{e,i} = \top$. From $I_i = \top$ we know that $V'_{e,i} = \top$ or that $V'_{e,i}$ is a valid contract for P_i. From $V_{e,i} = \top$ we know that $V'_{e,i} = \top \oplus \Delta_l \neq \top$. So, $V'_{e,i}$ is a valid contract. Since $V_{e,i} = \top$ and $V'_{e,i} = V_{e,i} \oplus \Delta_l$, it follows that $\top \oplus \Delta_l$ is a valid contract. Since P_C can efficiently compute Δ_l and knows $\top$, it follows that P_C can *efficiently* compute a valid contract for P_i after they input s_C. Since P_C were given no such contract, this would be a break of the signature scheme if it happened with more than negligible probability.

Assume then that $V_{e,i} = \sigma$ is a valid contract for P_i. Again, from $I_i = \top$ we know that $V'_{e,i} = \top$ or that $V'_{e,i}$ is a valid contract σ' for P_i. Assume that $V'_{e,i} = \top$. In this case it can again be seen that $\Delta_l = \top \oplus \sigma$, which implies that P_C could efficiently compute σ, a contradiction, as they are missing at least one share. Assume then that $V'_{e,i} = \sigma'$. Here $\Delta_j = \sigma \oplus \sigma'$. This implies that P_C given σ could efficiently compute σ'. Since $\Delta_j \neq 0$ it follows that $\sigma' \neq \sigma$. This means that P_C given σ could efficiently compute a new contract $\sigma' \neq \sigma$. The construction of contracts implies that this requires to break the signature scheme of all non-colluders. □

E.9 Hybrid 7

Hybrid 7 is essentially just a bookkeeping step. We fill in the changes introduced from Hybrid 5 to Hybrid 6 and move as many details from the communication device to the recommended strategy.

The communication device $\mathcal{C}^{\text{h7}}$ works as follows:

key distribution: For each P_i sample a key pair (vk_i, sk_i) and output (vk, sk_i) to P_i.

compute result: Each P_i inputs some b_i and some (vk', sk'_i). If all P_i input the same vk' and sk'_i which is a signature key for vk'_i, then sample $(j, p) \leftarrow \text{Mec}(b)$, $e_0 \leftarrow \text{Epoch}(E)$, $\sigma = \text{Contract}((j,p), sk')$, $(V_{1,1}, \ldots, V_{E,n}) \leftarrow \text{Values}(\sigma, (e_0, j), E)$, $(S_{1,1}, \ldots, S_{E,n}) \leftarrow \text{Sharings}(V)$. Let $y_i = (S^{(i)}_{1,1}, \ldots, S^{(i)}_{E,n})$, where $S^{(i)}_{e,j}$ is the share of P_i of $S_{e,j}$. Otherwise, let all $y_i = \perp$.

round-robin delivery: Deliver $(y_1, \ldots, y_n)$ to $(P_1, \ldots, P_n)$ in unfair round-robin.

release contract: Allow for En rounds of standard communication.

accept contract, side-channel: As $\mathcal{C}^{\text{h4}}$ respectively $\mathcal{C}_{\text{REJ}}$.

The recommend strategy π^{h7}_j for P_j is as follows:

1. Input $b_j = t_j$ and $(vk', sk'_j) = (vk, sk_j)$.
2. Use the standard recommended strategy for unfair round-robin, and afterwards initialize a variable $d_j \in \{\texttt{allegiance}, \texttt{defection}\}$, where $d_j = \texttt{defection}$ iff $\perp$ was received from the unfair round robin.
3. Use the En rounds of standard communication to sequentially run E epochs, each consisting of tries $i = 1, \ldots, n$. In each epoch e, try $i \neq j$ send $s_j = S^{(j)}_{e,i}$ to P_i if $d_j = \texttt{alligiance}$ and send $s_j = \perp$ to P_i otherwise. In epoch e, try j, compute $V_{e,j} = \oplus_{i=1}^{n} s_i$. If $V_{e,j} \neq \top$ and $V_{e,j}$ is not a valid contract, let $I_j = \perp$. Otherwise, let $I_j = \top$. If $I_j \neq \perp$ and $V_{e,j}$ is a valid contract for P_j, then input it to $\mathcal{C}^{\text{h6}}$. If $I_j = \perp$, then let $d_j = \texttt{defection}$.
4. If a valid contract on (j,p) was received, give the local output $w_j = (t_j, (j,p))$. Otherwise, give the local output $w_j = (t_j, \perp)$.

Figure 14: Hybrid 7

Theorem 11 *$(\mathcal{C}^{h7}, \pi^{h7})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h6}, \pi^{h6})$.*

Proof: Almost trivial as the step consists essentially of syntactic changes. The only real change is that P_i now has to re-input (vk, sk_i) to the device, or it will output $\perp$ to all parties in the unfair round-robin. Since at least one non-colluder will input $vk' = vk$, all colluders must do the same or have the device output $\perp$ to all parties in the unfair round-robin. Therefore all colluders must input sk'_i which is a valid signing key for pk_i or have the device output $\perp$ to all parties in the unfair round-robin. So, the only extra power given to colluders is that they can make the device output $\perp$ to all parties in the unfair round-robin. This power they already had, as they just input `hold` in all rounds of the unfair round-robin. Specifying the strategy mapping is straight forward. $\square$

E.10 Hybrid 8

In the last hybrid (Fig. 15) we replace the device's sampling of the y_i and their unfair round-robin delivery by a secure MPC protocol. We first define the notion of security that we need and a construction using standard MPC building blocks.

Definition 7 *Let $\mu = (\mu_1, \ldots, \mu_n)$ be an n-party protocol communicating via $\mathcal{C}_{\mathtt{int}}$. Let $(y_1, \ldots, y_n) \leftarrow f(\kappa, x_1, \ldots, x_n, r)$ be a randomized n-party function indexed by a security parameter κ. Let $C \subseteq \{1, \ldots, n\}$ specify a subset of* corrupted parties*, and let $H = \{1, \ldots, n\} \setminus C$ denote the* honest parties.

In the protocol setting*,* alternative programs *μ_C^* are specified for the corrupted parties, inputs $x = (x_1, \ldots, x_n)$ are specified and the protocol is executed to produce $(y_1, \ldots, y_n) \leftarrow \mu'(x)$, where $\mu' = (\mu_C^*, \mu_H)$ and y_i is the final local output of μ_i'. The alternative programs are allowed to communicate using the side-channel. We use $\textsc{Exec}_\mu(x, \kappa)$ to denote the distribution of the outputs y on inputs x and security parameter κ, and we use $\textsc{Exec}_\mu = \{\textsc{Exec}_\mu(x, \kappa)\}$ to denote the family of distributions indexed by x and κ.*

In the ideal setting*, a* simulator *$\mathcal{S}^C$ is given. Since the simulator is allowed to depend on μ_C^*, we sometimes write $\mathcal{S}^C(\mu_C^*)$. First $\mathcal{S}^C$ is run on x_C to produce* alternative inputs *x_C^*. Then $(y_1, \ldots, y_n) \leftarrow f(\kappa, (x_C^*, x_H), r)$ is sampled for a uniformly random r. Then $\mathcal{S}^C$ interacts in an unfair round-robin delivery of the y_i: The simulator $\mathcal{S}^C$ inputs `hold` or `send` on behalf of P_C and the parties P_H run the recommended strategy for unfair round-robin. This gives an output $y_i' \in \{0,1\}^* \cup \{\perp\}$ for $i \in H$. Then $\mathcal{S}^C$ outputs y_i' for $i \in C$, and the output of the simulation is $y' = (y_1', \ldots, y_n')$. We use $\textsc{Ideal}_{\mathcal{S}^C(\mu_C^*), f}(x, \kappa)$ to denote the distribution of the outputs y' on inputs x and security parameter κ, and we use $\textsc{Ideal}_{\mathcal{S}^C(\mu_C^*), f} = \{\textsc{Ideal}_{\mathcal{S}^C(\mu_C^*), f}(x, \kappa)\}$ to denote the family of distributions indexed by x and κ.*

We say that μ is a t-secure implementation of f with unfair round-robin delivery if for all C, $|C| \leq t$ there exists a poly-time simulator $\mathcal{S}^C$ such that $\textsc{Exec}_\mu$ and $\textsc{Ideal}_{\mathcal{S}^C(\mu_C^*), f}$ are computationally indistinguishable for all poly-time μ_C^*.

Theorem 12 *Under standard cryptographic assumptions, for all poly-time f, there exists an $(n-1)$-secure implementation of f with unfair round-robin delivery.*

Proof: One can use any generic secure MPC protocol with threshold $t = n - 1$ to compute verifiable secret sharings (VSS) $S_1, \ldots, S_n$, where S_i is a VSS of y_i with threshold $t = n - 1$. Any P_i which sees the generic secure MPC protocol fail outputs $y_i' = \perp$. Then in sequence, $i = 1, \ldots, n$, all P_j securely send their share of S_i to P_i — if P_j already output $y_j' = \perp$ it sends the share $\perp$. If any party P_j sent $\perp$, then P_i outputs $y_i' = \perp$; Otherwise P_i tries to reconstruct S_i to some y_i. If the reconstruction fails, then P_i outputs $y_i' = \perp$; Otherwise P_i outputs $y_i' = y_i$.

For the VSS one can e.g. use an additive secret sharing $y_i = \oplus_{j=1}^n S_i^{(j)}$ and give P_i the share $S_i^{(j)}$ along with a uniformly random field element $b_i^{(j)} \in F$ (where F is a finite field large enough to hold $S_i^{(j)}$). Then P_i is given a uniformly random field element $a_i^{(j)} \in F$ and $T_i^{(j)} = a_i^{(j)} S_i^{(j)} + b_i^{(j)}$. To send its share P_j sends $(S_i^{(j)}, b_i^{(j)})$ and P_i checks that $T_i^{(j)} = a_i^{(j)} S_i^{(j)} + b_i^{(j)}$. It is easy to see that P_j can have an incorrect share accepted by P_i with probability at most $1/|F|$: If P_j could send $S_i^{(j)\prime} \neq S_i^{(j)}$

and $b_i^{(j)\prime}$ such that $T_i^{(j)} = a_i^{(j)} S_i^{(j)\prime} + b_i^{(j)\prime}$, then it could compute $a_i^{(j)} = (b_i^{(j)\prime} - b_i^{(j)})(S_i^{(j)} - S_i^{(j)\prime})$, and $a_i^{(j)}$ is uniformly random in F in the view of P_j. The simulator can, on the other hand, make sure to know all the $a_i^{(j)}$-values, and can then exactly compute a correct $b_i^{(j)\prime}$ for all $S_i^{(j)\prime}$.

The simulator $\mathcal{S}^C$ then simulates a run of μ_C^* in the generic secure MPC protocol generating simulated VSS's $S_1', \dots, S_n'$ of 0, say. Then it runs the unfair round-robin part of the μ_C^*. If some μ_j^* sends an incorrect share, $\mathcal{S}^C$ inputs `hold` on behalf of P_j; Otherwise $\mathcal{S}^C$ inputs `send`. If $\mathcal{S}^C$ receives an output $y_j \neq \perp$ for P_j, then $\mathcal{S}^C$ changes the share $S_j^{(i)}$ of some honest P_i and changes it to $S_j^{(i)\prime} = S_j^{(i)} \oplus y_j$, and changes $b_j^{(j)\prime}$ to the corresponding value. Then it runs as in the protocol. □

The details of replacing mediation by the secure MPC protocol are given in Fig. 15.

The communication device $\mathcal{C}^{\text{h8}}$ works as follows:

key distribution: For each P_i sample a key pair (vk_i, sk_i) and output (vk, sk_i) to P_i.

compute result and release contract: Allow $A + En$ rounds of standard communication, where A rounds are enough to run a secure MPC protocol specified below.

accept contract, side-channel: As $\mathcal{C}^{\text{h4}}$ respectively $\mathcal{C}_{\text{REJ}}$.

The recommend strategy π_j^{h8} for P_j is as follows:

1. Receive and store (vk, sk_j).
2. Run the code of P_j in a secure MPC with unfair round-robin delivery for the following probabilistic function f:
   - Each P_i inputs some b_i and some (vk', sk_i'). If all P_i input the same vk' and sk_i' a signature key for vk_i', then sample $(j, p) \leftarrow \text{Mec}(b)$, $e_0 \leftarrow \text{Epoch}(E)$, $\sigma = \text{Contract}((j,p), sk')$, $(V_{1,1}, \dots, V_{E,n}) \leftarrow \text{Values}(\sigma, (e_0, j), E)$, $(S_{1,1}, \dots, S_{E,n}) \leftarrow \text{Sharings}(V)$, and let $y_i = (S_{1,1}^{(i)}, \dots, S_{E,n}^{(i)})$, where $S_{e,j}^{(i)}$ is the share of P_i of $S_{e,j}$. Otherwise, let $y_i = \perp$. The output to P_i is y_i.

   Use inputs $b_j \leftarrow B_j(t_j)$ and $(vk', sk_j') = (vk, sk_j)$.
3. Afterwards initialize a variable $d_j \in \{\texttt{allegiance}, \texttt{defection}\}$, where $d_j = \texttt{defection}$ iff the secure MPC protocol output $y_j \neq \perp$. If $d_j \neq \texttt{defection}$, then parse y_j as shares $(S_{1,1}^{(j)}, \dots, S_{E,n}^{(j)})$.
4. Use En rounds of standard communication to sequentially run E epochs, each consisting of tries $i = 1, \dots, n$. In each epoch e, try $i \neq j$ send $s_j = S_{e,i}^{(j)}$ to P_i if $d_j = \texttt{alligiance}$ and send $s_j = \perp$ to P_i otherwise. In epoch e, try j, compute $V_{e,j} = \oplus_{i=1}^n s_i$. If $V_{e,j} \neq \top$ and $V_{e,j}$ is not a valid contract, let $I_j = \perp$. Otherwise, let $I_j = \top$. If $I_j \neq \perp$ and $V_{e,j}$ is a valid contract for P_j, then input it to $\mathcal{C}^{\text{h6}}$. If $I_j = \perp$, then let $d_j = \texttt{defection}$.
5. If a valid contract on (j,p) was received, give the local output $w_j = (t_j, (j,p))$. Otherwise, give the local output $w_j = (t_j, \texttt{sorry})$.

Figure 15: Hybrid 8

Theorem 13 *$(\mathcal{C}^{h8}, \pi^{h8})$ is an $(n-1)$-resilient ϵ-Nash implementation of $(\mathcal{C}^{h7}, \pi^{h7})$.*

Proof: Q1 follows from the MPC being correct when all parties follow the code.

Strategy mapping. For properties Q2 and Q3 we must for each P_l, $l \in C$ and each strategy $\pi_l^{\mathrm{h8}*}$ for $\mathcal{C}^{\mathrm{h8}}$ specify a strategy $\pi_l^{\mathrm{h7}*}$ for $\mathcal{C}^{\mathrm{h7}}$.

The strategy $\pi_l^{\mathrm{h7}*}$ runs $\pi_l^{\mathrm{h8}*}$ with $\mathcal{C}^{\mathrm{h7}}$. In addition it lets $\pi_l^{\mathrm{h8}*}$ communicate with the $\pi_j^{\mathrm{h8}*}$ run by other $\pi_j^{\mathrm{h7}*}$, $j \in C$ as they would have done in Hybrid 8. It also uses the simulator $\mathcal{S}^C$ of μ.

The only modification needed is due to the fact that $\mathcal{C}^{\mathrm{h7}}$ takes an input (b_l, vk', sk'_l) and delivers an output y_l, whereas when using $\mathcal{C}^{\mathrm{h8}}$ the value y_l is computed using the secure MPC protocol. This is handled using $\mathcal{S}^C$. When $\pi_l^{\mathrm{h8}*}$ reaches the point where it is about to take part in the execution of μ, $\pi_l^{\mathrm{h7}*}$ lets $\pi_l^{\mathrm{h8}*,1}$ be the current state of $\pi_l^{\mathrm{h8}*}$. So, $\pi_l^{\mathrm{h8}*,1}$ is in particular a piece of code ready to run the part of P_l in $\mu(x)$. Define $x_C = \{x_l\}_{l \in C}$ to be the inputs the parties P_C would input to $\mu(x)$. Note that P_l, $l \in C$ knows $x_l = (b_l, vk', sk'_j)$ as they were output by $\pi_l^{\mathrm{h8}*}$, as it thought it was sending them to $\mathcal{C}^{\mathrm{h8}}$. Let μ_l^* be a piece of code which runs $\pi_l^{\mathrm{h8}*,1}$ and in the end outputs the state of $\pi_l^{\mathrm{h8}*,1}$ after this execution, which we call $\pi_l^{\mathrm{h8}*,2}$. Let $\mu_C^* = \{\mu_l^*\}_{l \in C}$.

Now the parties $P_l \in P_C$ send x_l and μ_l^* to a coordinator, e.g., the lowest indexed colluder. The coordinator runs $\mathcal{S}^C(\mu_C^*)$ on input x_C to get alternative inputs x_C^*, and sends x_l^* to $\pi_l^{\mathrm{h7}*}$, which inputs x_l^* to $\mathcal{C}^{\mathrm{h7}}$. Then the coordinator runs $\mathcal{S}^C(\mu_C^*)$ with the unfair round-robin delivery of $\mathcal{C}^{\mathrm{h7}}$. If $\mathcal{S}^C(\mu_C^*)$ sends $s_j \in \{\texttt{send}, \texttt{hold}\}$ on behalf of P_j, $j \in C$, then the coordinator sends s_j to P_j which inputs s_j to $\mathcal{C}^{\mathrm{h7}}$. When $\mathcal{C}^{\mathrm{h7}}$ delivers y_j to P_j, it sends it to the coordinator which inputs it to $\mathcal{S}^C(\mu_C^*)$. In the end $\mathcal{S}^C(\mu_C^*)$ outputs $\{y'_j\}_{j \in C}$, where y'_j simulates the final output of μ_j^*. Recall that the output of μ_j^* is the state of $\pi_l^{\mathrm{h8}*,1}$ after an execution of unfair round-robin delivery, which we called $\pi_j^{\mathrm{h8}*,2}$. Therefore y'_j is a simulation of $\pi_j^{\mathrm{h8}*,2}$. The coordinator sends $\pi_j^{\mathrm{h8}*,2}$ to each P_j. After the execution of the MPC protocol respectively the unfair round-robin delivery of $\mathcal{C}^{\mathrm{h7}}$, Hybrid 8 and Hybrid 7 are identical. So, now each $\pi_j^{\mathrm{h7}*}$ can finish the execution by letting $\pi_j^{\mathrm{h8}*,2}$ run with $\mathcal{C}^{\mathrm{h7}}$. When $\pi_j^{\mathrm{h8}*,2}$ makes a local output, $\pi_j^{\mathrm{h7}*}$ adopts this as its own local output.

As a final technicality, we ensure that the simulated states $\pi_j^{\mathrm{h8}*,2}$ include the sequence of contracts input to the device, for the following use: If the simulator sees that any simulated $\pi_j^{\mathrm{h8}*,2}$ contains a simulated inputting of a valid contract σ to $\mathcal{C}^{\mathrm{h8}}$, then for the first P_j to give such an input in the simulation, the coordinator informs P_j of its being first. Then on receiving $\pi_j^{\mathrm{h8}*,2}$ from the coordinator, P_j inputs σ to $\mathcal{C}^{\mathrm{h7}}$. Since no non-colluder inputs contracts in this phase of the execution, this possibly delayed inputting of σ will not change the identity of the *first* contract input to the device in the two settings.

Q3: In Hybrid 8, let D^{h8} denote the state of all parties after the execution of the MPC protocol. For π_j^{h8}, $j \notin C$, this is t_j, vk and the output y_j from the MPC protocol — the party has to remember t_j to output $o_j = t_j$, has to remember vk to verify possible contracts and besides this only has to remember y_j. For $\pi_j^{\mathrm{h8}*}$, $j \in C$, the state is some value, which we call $\pi_j^{\mathrm{h8}*,2}$.

In Hybrid 7, let D^{h7} denote the state of all parties after the execution of the MPC protocol. For π_j^{h7}, $j \notin C$, this is t_j, vk and the output y_j from the unfair round-robin delivery. For $\pi_j^{\mathrm{h7}*}$, $j \in C$, we let it be the value $\pi_j^{\mathrm{h8}*,2}$ sent by the coordinator.

By construction of the coordinator and the $(n-1)$-security of the MPC protocol, the distributions D^{h8} and D^{h7} are computationally indistinguishable. In both hybrids the local outputs are computed using the same poly-time computation on D^{h8} respectively D^{h7}, namely completion of the protocol with $\mathcal{C}^{\mathrm{h8}}$ respectively $\mathcal{C}^{\mathrm{h7}}$, which are identical throughout the completion of the protocol. Since poly-time computation maintains computational indistinguishability, it follows that the local outputs are computationally indistinguishable.

Q2: Using the same argument as above, the distribution of the first valid contract input to the device is computationally indistinguishable in the two hybrids — it is the same poly-time function of $D^{\mathtt{h8}}$ respectively $D^{\mathtt{h7}}$. $\square$

E.11 Concluding

Since the unmediated protocol (Fig. 5) is identical to Hybrid 8, Theorem 3 follows from Theorem 4 to Theorem 13.