On Cryptographic Cheap Talk with Smart Contracts

Pavel Hubáček[1,2] and Michelle Yeo[3,4]

[1] Institute of Mathematics, Czech Academy of Sciences
[2] Charles University, Faculty of Mathematics and Physics
[3] Nanyang Technological University
[4] Aarhus University

Abstract. In this work, we revisit cryptographic cheap talk, i.e., preplay protocols for implementing correlated strategies in strategic games, and study its power in a new setting where we allow the players to additionally interact with smart contracts. As our main result, we present a protocol that uses a smart contract as a correlation device to publicly sample and publish an action profile from any distribution over the set of Pareto optimal action profiles. We show that players are not only incentivised to follow this public advice, but are also incentivised not to renegotiate to any other strategy when the sampling distribution is additionally Pareto optimal. We also propose a novel model for analysing smart contract extortion attacks in this setting. We analyse these extortion attacks in our protocol and show that, when action profiles are further restricted to give both players the same payoff, our protocol is safe from smart contract extortion attacks. Our ideal protocol uses simultaneous broadcast channels, which are difficult to implement in practice. We propose a method to bypass the need for simultaneous broadcast using symmetric key encryption and time-lock puzzles.

Keywords: Cheap talk · Smart contracts · Blockchains · Nash Equilibrium · Renegotiation-safety

1 Introduction

A *correlated equilibrium* [7] in an n-player normal-form strategic game is a joint strategy μ for the players, i.e., a probability distribution on action profiles, with the stability property that, upon privately receiving its action a_i in an action profile $(a_1, \ldots, a_n)$ sampled from μ, no player has an incentive to deviate from playing a_i to some other action assuming that the others will play according to the sampled action profile. The common interpretation of correlated equilibrium is that the players have access to some *correlation device* sampling action profiles according to μ and delivering the respective action a_i privately to each player i as a suggested advice for their play. From the computational perspective, correlated equilibrium has the advantage that it is efficiently computable via linear programming [41]. However, the necessity of a trusted correlation device makes it

rather cumbersome from the practical implementation perspective. In principle, one can restrict the possible strategies to product distributions, i.e., avoiding the need for the correlation device as each player could then sample its component of the action profile independently. However, the solution concept would then lose its computational appeal as, for product distributions, correlated equilibria collapse to the set of Nash equilibria, which are not known to be efficiently computable in general and, in fact, are PPAD-hard even to approximate [18].

The question of equilibrium implementation has been studied by various communities. In the game-theoretic literature, many works studied preplay protocols for correlating player's strategies prior to play in normal-form games (see, e.g., [8] and the references therein). A particular flavour of such extensions of simultaneous-move games is the so-called *cheap talk*, i.e., a version of the game where, before taking an action, the players can exchange an unbounded number of messages that are unmediated, nonbinding, and payoff-irrelevant. It was shown by Bárány [11] that any set of at least four players can implement an arbitrary correlated equilibrium via cheap talk. Later, cryptographic techniques from secure multiparty computation were leveraged to implement any correlated equilibrium of an arbitrary normal-form game (i.e., without the restriction on the number of players) assuming computationally bounded players [19]. Though, all these feasibility results are arguably far from the inception of cheap talk consisting of "plain face-to-face conversations".

In more detail, when implementing a correlated equilibrium via (cryptographic) cheap talk, the players need to devise a method for sampling a correlated action profile and distributing the respective parts of the action profile to the parties privately so that none of them sees the opponents' advice. Clearly, a trusted mediator can easily perform both the sampling of the correlated action profile and, assuming the existence of private channels, the distribution of the private advice. On the other hand, a preplay protocol run solely among the players must resolve both of these non-trivial issues at once, resulting in constructions employing the full power of secure multi-party computation. In fact, as shown in [26], such preplay protocols implementing interesting classes of correlated equilibria imply secure protocols for "complete" primitives for secure computation such as oblivious transfer and, thus, are necessary. In this work, we revisit the problem of implementing correlated strategies via *simple* preplay protocols and ask the following question:

Is there a realistic model without trusted mediation that allows implementing or extending some non-trivial class of correlated equilibria without relying on the full power of secure computation?

Our approach to the above question is to leverage the power of smart contracts on some blockchain. Importantly, smart contracts are stored in a distributed manner on some blockchain and their code is publicly observable. The introduction of smart contracts shifts the core considerations when designing a preplay protocol. In particular, a smart contract allows the parties to deposit a collateral prior to play that can be governed algorithmically. Consider, for example, the following simple protocol for implementing a correlated equilibrium μ

of some two-player normal-form game Γ. First, each player deposits a collateral into the smart contract. Then, *both players observe a complete action profile* sampled from μ upon which they can place their actions in Γ. After the gameplay in Γ, the players get their collateral back only if they played according to the sampled advice from μ. On a high level, each player deposits the collateral to "self-incentivize" to play according to the public advice. Clearly, it might be beneficial for a player to deviate upon learning the *whole* action profile, yet the possibility of losing the collateral deposited earlier allows to enforce the sampled action profile.

Public sampling of action profiles. An important difference to classical correlation devices for implementing correlated equilibria from the game-theoretic literature is that the above approach using a smart contract sidesteps the need for *private* sampling of the advice. In principle, we limit the set of correlated equilibria that can be implemented in this manner as the stability of the solution concept assumes that the advice for each player is private and does not give any guarantees otherwise. Though, it is a priori not clear how much the set of attainable distributions shrinks because of the effects of the collateral deposits. Most importantly, the main advantage of public sampling of the whole action profile is that it opens up new means of sampling the correlated advice for the players and is *significantly easier* to implement, e.g., via the smart contract itself or by leveraging public randomness beacons (see also Section 4).

Non-observable gameplay. On the other hand, a central assumption in the above simple protocol is that the gameplay is publicly observable and the settlement of collateral can be performed by the smart contract instead of a trusted third party. Clearly, assuming that a smart contract has a way of learning the outcomes of an arbitrary game in the real world is unrealistic. In fact, this is a fundamental issue hindering meaningful applicability of smart contracts when interfacing with the physical world or, more broadly, information from outside their native blockchain. For example in the context of systems for e-commerce using cryptocurrencies (e.g., BitHalo [54]), this is referred to as the *fair exchange problem*. Interestingly, a game-theoretic analysis of the fair exchange problem was given in [22] who showed that it cannot be resolved solely via smart contracts without trusted attestation of actions of the buyer and seller.

Given the importance of the scenario where the smart contract is completely oblivious to the outcomes of the game, we insist on mechanisms where any resolution of the collateral must be done solely based on inputs submitted to the smart contract *by the players*. Nevertheless, a protocol that necessitates both players to agree on the history of the play in Γ might easily run into an unsolvable deadlock. Our crucial observation, however, is that there is an important subset of correlated distributions the players can meaningfully implement in this setting, as we discuss next.

Pareto optimal action profiles. In a setting where communication between players is allowed and the sampled action profile is public, it will be played by the players only if it is Pareto optimal (i.e., when neither party can improve their utility without making the other worse off) – otherwise, the players could rene-

	Chicken	Dare
Chicken	$(6,6)$	$(2,7)$
Dare	$(7,2)$	$(0,0)$

Table 1: Payoff matrix for the game of chicken.

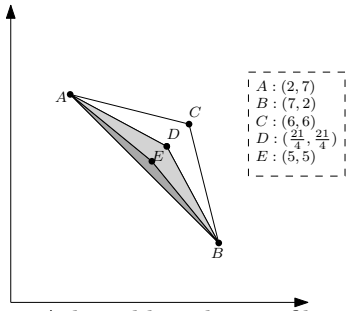

Fig. 1: Achievable utility profiles of Nash equilibria, correlated equilibria and Pareto optimal strategies for Chicken.

gotiate and jointly switch to some action profile that makes at least one of them better off. Thus, our proposed protocol restricts the distribution μ to distributions that sample from the subset of Pareto optimal action profiles. An example of Pareto optimal action profiles in the game of Chicken with payoff matrix depicted in Table 1 are the profiles (C, C), (D, C), and (C, D) (corresponding to utility profiles $(6, 6), (7, 2)$, and $(2, 7)$ respectively). Note that the restriction to Pareto optimal strategies is also useful from the perspective of social welfare as the Pareto optimal frontier of these strategies can never be dominated by the Pareto optimal frontier of correlated equilibria, as depicted in the Chicken example in Figure 1. The dark grey triangle in Figure 1 depicts the utility profiles achievable in the convex hull of the Nash equilibria, and the light grey extension of the dark grey triangle depicts the extended utility profiles that correspond to the convex hull of the correlated equilibria. All utility profiles on the lines AC and CB in Figure 1 correspond to Pareto optimal strategies. See Section 2.3 for additional discussion on the relationship of (coarse) correlated equilibria and distributions supported on Pareto optimal action profiles.

1.1 Our results

Our protocols. The protocols we propose in our work are extremely simple. First, we propose a protocol that works in an ideal setting with simultaneous broadcast channels. In this ideal protocol, the players and the smart contract only exchange four messages. All computations are done by the smart contract and only involve additions and comparisons. The only potential drawback is that we rely on simultaneous broadcast channels to send one of the messages from the players to the contract, and these channels are notoriously problematic to implement in practice. We therefore propose an alternative protocol that uses symmetric key encryption and time-lock puzzles to replace simultaneous broadcast channels, at the cost of some computational burden to the smart contract. We also stress that none of our protocols involve any trusted external party.

This gives our protocol the advantage of being actually practical and deployable in untrusted and distributed settings. Finally, we also show that all our results from our ideal protocol transfer computationally to our protocol with time-lock encryption. Specifically, we establish the following:

1. Our main theorem (Theorem 1) establishes that, for any two-player normal form game, any distribution from the set of distributions supported on Pareto optimal action profiles can be implemented as a renegotiation-safe subgame perfect equilibrium of the corresponding cheap talk extension with smart contracts. Our approach (detailed in Section 2.1) eliminates the need to send private advice to each player individually when implementing correlated equilibria and, hence, the need for private communication channels.
2. We propose a protocol using symmetric key encryption and time-lock puzzles to replace simultaneous broadcast in our ideal protocol. We show in Theorem 2 that our results in the ideal setting also transfer to this setting if we adopt the computational variants of Nash Equilibria and renegotiation-safety.

Comparing to MPC-based protocols. While MPC can theoretically implement any correlated equilibrium, it introduces significant practical and theoretical limitations compared to our smart contract approach. First, regarding trust models: MPC protocols typically require strict assumptions about the adversary, such as a semi-honest model or an honest majority threshold (e.g., assuming less than half or third of players are corrupt). In the anonymous, permissionless setting of blockchains, these assumptions are often unrealistic. In contrast, our smart-contract-based protocol operates under a full rationality model. We do not require players to be "honest"; we only require them to be rational agents maximizing their financial utility, a more robust assumption for DeFi environments. Second, regarding efficiency: MPC imposes a heavy communication and computational overhead. Implementing the complex circuit logic required to sample from arbitrary probability distributions via MPC requires multiple rounds of communication between all parties. This overhead becomes prohibitively expensive on-chain or even off-chain when latency is critical. Our approach offloads the coordination complexity to a simple, publicly verifiable smart contract, acting effectively as a "fire-and-forget" correlation device with minimal gas costs and zero interactive communication rounds during the sampling phase.

A model for analysing extortions. A key advantage when using smart contracts for arbitration is that not only can one design a smart contract to respond in any way given some input, one can also ensure that the response is *binding*. In the context of games with deposit schemes, this implies that players can be assured that any misbehaviour, if reported to the contract, will be punished. This feature of smart contracts can be, however, a double edged sword as players can employ their own smart contracts to launch binding extortion threats on the other player, especially if they control the other player's collateral. This potentially exposes our protocol to new avenues of attacks, and thus in our work we also consider a particular model of extortion threats using smart contracts and discuss mitigations to these threats. We show that if we firstly assume

strong identities, any extortion attempt using smart contracts leaves a trail of undeniable digital evidence due to the public and binding nature of these smart contracts. Additionally, if we assume an external legal system with corresponding legal fees and penalties, the extortion amount is upper bounded by the legal fees. We also highlight a particular counter extortion attack that any smart contract extortion attempt is vulnerable to: if the extortioner and their victim begin with the same amount of funds, the victim can always successfully counter the extortion threat. An important implication of these results is that in the one-shot setting, our protocol is additionally safe from smart contract extortions if we further restrict the distribution μ to only sample from action profiles which give players the same payoff. Specifically, we establish the following:

1. In Section 3.2, we present a novel model and analysis of extortion attacks in games with deposit schemes using smart contracts. We use the results in this section to additionally show in Proposition 1 that our protocol is safe from smart contract extortion attacks in the one-shot setting if we only sample action profiles which give both players the same payoff.

Assumptions and limitations. Below, we highlight the main assumptions in our model and limitations of our protocols.

1. Our protocols implement only distributions supported over Pareto optimal action profiles. Looking ahead, we use this to prove Lemma 2 that following the sampled advice is renegotiation-safe.
2. Our model and analysis of extortions using smart contracts assume strong identities to ensure that the extortions are undeniable. We also assume the existence of an external judicial system together with extortion penalties and legal fees to analyse the counter extortion attacks in Lemma 5 and Lemma 6. We note that a similar assumption on the existence of an external judicial system was explored in the cryptographic line of work on security against *covert adversaries* [9,23,17].

1.2 Related work

Cheap talk. An introductory motivation of cheap talk is given in [20], see also [8] for an extensive discussion of related work in this area. Trusted mediation to improve the utility of agents in multiagents systems have been studied in [7,5,30]. Besides the independent work of [11], several works drew explicitly on techniques from cryptography to eliminate the trusted third party as a mediator when implementing correlated equilibria. [19] proposed a two-party protocol that implements the correlated element selection problem using blindable encryption. Subsequently, [48] and [6] built on this line of work and improved its efficiency. [24] and [26] considered sequential rationality of such cryptographic implementations and [27] extended these techniques to implement *coarse* correlated equilibria. In the blockchain setting, [25] implements correlated equilibria transaction pricing mechanisms to optimise transaction fees for users. Similar to

our goal, [10] replaces trusted mediation with smart contracts, but relies on heavier cryptographic tools like oblivious transfer to ensure that the sampled advice remains private. Finally, the line of work in [37,28,29] studied a strong notion of implementation achievable by physical devices such as secure ballot-boxes.

Incentivising behaviour via cryptocurrency deposits. Most related to our results are the works of [44,45] who studied the power and computational aspects of "payment schemes" for incentivising behaviour of players in strategic games. The crucial difference is that, in his model, there is a way of inferring the outcome of the game probabilistically, and this can be used by a smart contract in order to fairly dole out punishments and rewards to the players. Note that [45] explicitly studied only implementation of *deterministic* strategies but his work is in principle applicable also to randomised strategies under the assumption that there is some means of inferring the outcome of the game play. Crucially, in this work, we study the challenging case not covered by [45] when the smart contract (or payment scheme) cannot infer anything about the outcome of the game and present a protocol where the punishment mechanism is simply a consequence of players observing each other and responding accordingly.

Prior to [45], various works in the cryptographic literature explored the idea of using cryptocurrency collateral deposits for incentivizing honest behaviour in protocols for secure computation, e.g., for implementing secure distributed lotteries or poker [13,36,14], achieving fairness in secure computation [3], and, more recently, for covert security [9] in general secure computation [53,21].

Blockchain fraud proof systems. Fraud proof systems [49,47,32,2] operate on the principle of verifying computational correctness, ensuring that state transitions adhere strictly to the deterministic rules of the protocol. In such systems, a dispute is resolved by proving that a specific sequence of operations yields a state different from what was published. However, this framework is insufficient for resolving the *smart contract extortion attacks* we analyze. The verification required in our setting is not whether a computation was executed correctly, but whether an extortion took place. An extortion attempt relies on an auxiliary smart contract that commits to a strategy (e.g., burning funds if a demand is not met). Crucially, this auxiliary contract executes valid code and produces a valid blockchain state; there is no "incorrect computation" for a fraud proof to challenge. Furthermore, such contracts can be disguised as legitimate financial instruments, making algorithmic detection impossible without context. Consequently, mitigating these threats requires an *external judicial system* capable of establishing *mens rea* (intent) and analyzing the sequence of events to judge the counterparty's behavior, a task that lies beyond the scope of deterministic, state-based fraud proofs.

Timed-encryption. Further upfield but also related are works that utilise timed-encryption [43] to bypass simultaneous broadcast, in the general setting [4] as well as specific settings like auctions [52] and front-running prevention [51].

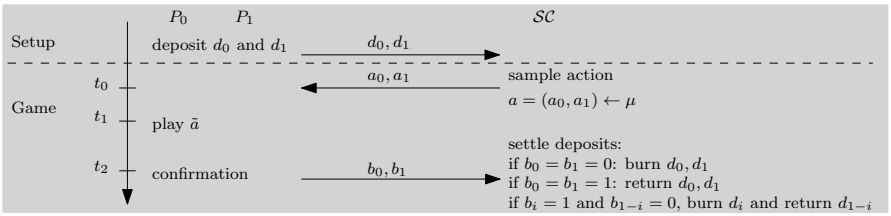

Fig. 2: Timeline depicting the sequence of events in $\Pi_2^{P.O.}$.

2 Our protocol

In this section, we present our protocol $\Pi_2^{P.O.}$. $\Pi_2^{P.O.}$ involves two players who wish to play an underlying strategic game Γ with a deposit scheme and a smart contract. After playing the underlying game Γ, both players additionally simultaneously input a confirmation bit. As our protocol induces a *two round extensive-form game*, i.e., a game that is played over two rounds, the solution concept we are interested in is the *subgame perfect equilibrium*. We defer all formal game theoretic definitions and solution concepts to Appendix A.1.

2.1 Protocol description

Notation. For $N \in \mathbb{N}$, we denote by $[N]$ the set $\{1, \ldots, N\}$. For a set S, we denote the set of all probability distributions on S by $\Delta(S)$. For a probability distribution $\mu \in \Delta(S)$, we denote by $x \leftarrow \mu$ sampling x from S according to μ. By $x \leftarrow S$, we denote sampling an $x \in S$ uniformly at random. We denote the support of μ by $\mathrm{supp}(\mu)$.

Let P_0 and P_1 denote two players who wish to play a two-player normal-form game $\Gamma = (A = (A_1, A_2), u_1, u_2)$. Let $\mathcal{SC}$ denote a smart contract. For $i \in \{0, 1\}$, we define $d_i = \max_{a \in A}(u_i(a)) - \min_{a \in \mathrm{supp}(\mu)}(u_i(a)) + 1$ to be the collateral that P_i has to deposit into $\mathcal{SC}$. Our protocol $\Pi_2^{P.O.}$ consists of two phases: a *setup* and a *game* phase, described in Figure 3 in Appendix B.1 and depicted in Figure 2.

Setup phase. The purpose of the setup phase is for both players to deposit their collateral into the smart contract $\mathcal{SC}$ and, additionally, to agree on three times, $t_0 < t_1 < t_2$ in the next phase of the protocol with the following significance. The time t_0 is when the smart contract publicly samples an advice action profile $a \leftarrow \mu$. The time t_1 is the time of play in the game Γ. The final time t_2 is when the deposited collateral gets settled.

Game phase. The game phase of the protocol begins at time t_0 where an action profile is publicly sampled, i.e., $a \leftarrow \mu$. After observing the public signal a, the two players actually play the game Γ they intend to play. We stress that the play in Γ is, in principle, unrelated to the protocol and the protocol is oblivious to whether the play occurs on the blockchain or in the real world. Finally, the

game phase concludes with a resolution of the collateral. Both players simultaneously send a confirmation bit to the smart contract that will determine whether the other player will get back their collateral. We note that simultaneity of confirmation is necessary to prevent a player from incurring any potential disadvantage from sending their confirmation message first. This is especially crucial when we analyse $\Pi_2^{P.O.}$ in the context of smart contract extortion attacks in Section 3.3 to ensure the extortioner cannot use the victim's confirmation action to threaten the victim (which can happen if the confirmations were sequential with the victim confirming first).

If both players confirm that the other player played according to the proposed action profile a at time t_1, both players will get back their deposit. If any player reports misbehaviour from the other player, the entire deposit of the other player is burnt. Here, we stress that a common slashing mechanism whereby both players' deposits are slashed when a player reports misbehaviour does not make sense in our setting as this would simply give players a free pass to misbehave. This is because they know the other player would not report their misbehaviour as it would result in the reporting player losing their deposit.

2.2 Analysing $\Pi_2^{P.O.}$ as a sequential game

In this section, we analyse the properties of $\Pi_2^{P.O.}$. In our modeling, we use the following simplifying assumptions, which we address and discuss in Section 4:

1. Perfect and trusted sampling of the action profile from μ.
2. Simultaneous confirmations at time t_2 from the players to the smart contract.

At first, we also additionally assume that the players cannot communicate but we lift this assumption later in this section. Under the above three assumptions, given a two-player normal-form game $\Gamma = (A = (A_1, A_2), u_1, u_2)$ and a distribution $\mu \in \Delta(A)$ over the set of action profiles A in Γ, our protocol $\Pi_2^{P.O.}$ induces a two-player, two-round extensive form game Γ' defined by the game-tree in Figure 4 in Appendix B.1. At the start of the game, both players receive an action profile $a = (a_0, a_1) \leftarrow \mu$ sampled from μ. In the first round of Γ', the players take a simultaneous move corresponding to playing the underlying normal-form game Γ. In the second round of Γ', both players simultaneously play their respective confirmation bits given the history h after the first round.

We proceed to demonstrate that it is rational for the players to follow the protocol. Let ν denote the strategy in Γ' corresponding to following $\Pi_2^{P.O.}$.

Definition 1. *Let Γ' be the game in Figure 4. We define ν to be the following strategy profile:*

First round: *Both players play according to the received action profile a.*
Second round: *Given the history $h = (h_0, h_1)$ after the first round, each player P_i plays confirmation bit 1 if $h_{1-i} = a_{1-i}$ and 0 otherwise.*

We first establish in the following lemma that players have no incentive to make private, unilateral deviations from the prescribed strategy ν. The proof of Lemma 1 is presented in Appendix E.1.

Lemma 1. *Let $\Gamma = (A_1, A_2, u_1, u_2)$ be a normal-form game and $\mu \in \Delta(A)$ a distribution of action profiles. The strategy profile ν from Definition 1 is a subgame perfect equilibrium in Γ'.*

Equilibria in the presence of communication – renegotiation-safe strategies. Importantly, the statement of Lemma 1 is rather strong as it makes no restriction on the quality of the distribution μ in terms of the guaranteed social welfare. However, many distributions μ would never be followed and implemented by the players if we lift the simplifying assumption that players do not communicate and consider a more expressive model where players can renegotiate to jointly deviate. Specifically, if the players receive an action profile a that is dominated by some action profile a' for both of them then they would jointly deviate to playing the more profitable action profile a' and submitting the confirmation bits $b_0 = b_1 = 1$ to retrieve the collateral deposits. Next, we proceed to formally analyze this setting with joint deviations. We show that, when restricting μ to the class of distributions sampling only Pareto optimal action profiles (defined in Appendix A.2), the players still have no incentive to deviate from the strategy ν even when they have the option of joint deviations.

To this end, we use a notion of renegotiation-safety inspired by Pass and Shelat [42]. We provide more details on how our definition differs from that of [42] in Appendix B.2. Let Γ be a two-player simultaneous extensive-form game. Formally, a *renegotiation* at some round in Γ is a publicly announced change from strategy profile σ to σ' that is to be played by both players at the given round and forth. We use $[\sigma, r, \sigma']$ to denote the strategy profile that consists of following σ for all rounds *before* round r and following σ' from r.

Definition 2. *(Renegotiation-safe strategies in games with simultaneous moves) Let Γ be an extensive-form game, where, at each round, the players make a simultaneous move. A strategy profile σ is* renegotiation-safe *at round r in Γ if for every player i there is no other σ' such that*

1. *$u_i([\sigma, r, \sigma']) > u_i(\sigma)$ and $u_j([\sigma, r, \sigma']) \geq u_j(\sigma)$ for all other players $j \neq i$, and*
2. *$[\sigma, r, \sigma']$ is renegotiation-safe at every round $r' > r$.*

We say a strategy profile is renegotiation-safe *if it is renegotiation-safe at all rounds in Γ.*

Next, we formally show in Lemma 2 (with proof in Appendix E.2) that ν is an equilibrium even under public renegotiations as long as the proposed action profiles are sampled from the set $\bar{A}$ of Pareto optimal action profiles in Γ.

Lemma 2. *For any $\mu \in \Delta(\bar{A})$, the strategy profile ν is renegotiation-safe in Γ'.*

Together, Lemmas 1 and 2 immediately give our main theorem below.

Theorem 1. *Let $\Gamma = (A_1, A_2, u_1, u_2)$ be a two-player normal-form game and $\mu \in \Delta(\bar{A})$ any distribution on the set of Pareto optimal action profiles in Γ. Then, in the two-round extensive-form game Γ' induced by $\Pi_2^{P.O.\cdot}$, the strategy ν of playing according to the advice sampled from μ and confirming honestly in round 2 of Γ' is a renegotiation-safe subgame perfect equilibrium of Γ'.*

2.3 Discussion

Here, we discuss and motivate sampling from distributions over Pareto optimal action profiles, give examples of how $\Pi_2^{P.O.\cdot}$ can be used to play Chicken and coordinate votes for decentralised voting. We consider extensions from the one-shot to the repeated setting, as well as to multiple players in Appendix B.3.

Distributions over Pareto optimal action profiles vs. (coarse) correlated equilibria. At this point, a careful reader surely observed that a distribution supported on Pareto optimal action profiles is not necessarily a correlated equilibrium. In fact, the two concepts are incomparable – in many games, there exist correlated equilibria that are not supported on Pareto optimal action profiles (e.g., any correlated equilibrium in the game of Chicken from Table 1 sampling the action profile (D, D) with positive probability.) Even though our work is motivated by the line of work on implementing correlated equilibria, the move to Pareto optimal action profiles seems unavoidable as we rely on *public* sampling of the complete action profile by a smart contract. On the other hand, such distributions are clearly of interest from the perspective of social welfare; the Pareto optimal frontier of distributions over $\bar{A}$ might coincide with the Pareto optimal frontier of correlated equilibria but it can never be dominated by it. In other words, rational players should prefer implementing the Pareto optimal frontier of $\Delta(\bar{A})$ over the Pareto optimal frontier correlated equilibria.

Additionally, there is a clear advantage of distributions from $\Delta(\bar{A})$ over correlated equilibria from the computational perspective. It is known that, for any normal-form game, *some* correlated equilibrium (unlike a Nash equilibrium) can be found efficiently [41]. However, as shown by Barman and Ligett [12], the task of optimizing interesting objective functions over the set of correlated equilibria such as social welfare is NP-hard even if we relax the equilibrium notion to the broader class of *coarse* correlated equilibria [39]. Compare this to optimizing objective functions over $\Delta(\bar{A})$, i.e., distributions expressible as convex combinations of action profiles from $\bar{A}$. Thus, the optimization problem reduces to computing the set of Pareto optimal action profiles, which can be performed naively in quadratic time for any bimatrix game.

Finally, note that, as players can efficiently optimize over $\Delta(\bar{A})$ in practice, any μ implemented by rational players using $\Pi_2^{P.O.\cdot}$ would lie on the Pareto optimal boundary of $\Delta(\bar{A})$. This is not captured in Theorem 1 due to the fact that we do not model the selection of μ as μ is treated as an input to $\Pi_2^{P.O.\cdot}$.

Applying our protocol to the game of Chicken. We highlight an example of how players can use $\Pi_2^{P.O.\cdot}$ to play the game of Chicken with payoff matrix depicted in Table 1. First note that the set of Pareto optimal distributions of Chicken lie on the Pareto frontier of the achievable utility profiles (i.e., on the lines AC and CB in Figure 1). In particular, the distribution μ that puts all its mass on (C, C) gives both players an expected utility of 6, which is larger than $\frac{21}{4}$, the expected utility of the correlated equilibrium. To play Chicken with μ using $\Pi_2^{P.O.\cdot}$, players first deposit $2+7+1 = 10$ coins as collateral into the smart contract. During the game phase, the smart contract publicly samples (C, C) and the players each play an action. When both players play according to (C, C)

and send confirmation bits of $b = 1$ during the confirmation time, the smart contract will return both deposits and the utility of both players would be 6. In the case where one player, wlog P_1, deviates and plays D at game time, P_0 will send a confirmation bit of $b = 0$. The smart contract burns the entirety of P_1's deposit in this case, leading to a final payoff of 2 for P_0 and -3 for P_1.

Bach-Stravinsky. An interesting application of our protocol in decentralised voting systems would be to coordinate votes for decentralised voting [1,34] or vote delegation [15,31], similar to the classic coordination game of Bach-Stravinsky. In this setting, we have two players P_0, P_1 that may have different preferred voting candidates (say candidate C_0 for P_0 and C_1 for P_1), but derive greater utility if they coordinate their votes and vote for either of their preference[5]. Our protocol can be modified in this setting to output a public, seemingly unrelated signal instead of an action profile (e.g., a bit b and P_0 and P_1 will vote for candidate C_b). Note that in this setting, players only have to decide on the mapping of the values of the public signal to candidates. After this decision is made, they do not need any further interaction in order to coordinate their votes.

3 The potential and limitations of smart contracts

As our protocol $\Pi_2^{P.O.}$ involves smart contracts, players can also create and employ their own smart contracts over the course of the protocol duration, which potentially leaves $\Pi_2^{P.O.}$ vulnerable to *smart contract attack vectors*. In particular, since players control the other player's deposit in round 2 of Γ', there is a risk in $\Pi_2^{P.O.}$ that players can employ smart contracts to launch extortion threats. In a two-player game, extortion threats are attacks where one player holds on to something (e.g., physical goods, digital collateral etc.) from the other player and only returns it if the other player agrees to some demands (e.g., play a specific action). In our setting, any player in control of the deposit of the other party can use the controlled deposit to make an extortion threat. We say an extortion threat is *effective* if the extortioner can prove to the victim that what the extortioner holds on to will be returned to the victim if the victim complies with the threat. An extortion threat is *successful* if the extortioner manages to force the victim to yield to their threat.

In the following subsections, we focus on analysing the power and limitations of smart contracts in our setting of two-player games with deposit schemes, with the ultimate goal of understanding the extent of the impact of these new attack vectors on our protocol. We begin by modelling extortions in this setting by a 2 player extortion game in Section 3.1, and then show in Section 3.2 that smart contracts can be employed in this setting to increase the power of extortions by making these threats effective. However, this comes at a price: there is now

[5] This makes most sense in a setting where there are several voters and candidates, and both players do not mind the other candidate option and strongly prefer them to the rest of the candidates. That is, P_0 prefers C_0 to C_1, but *strongly prefers* either C_0 or C_1 to C_i for $i > 1$.

public, undeniable digital evidence of the extortion. Finally, we return to our main protocol $\Pi_2^{P.O.}$ and show that under certain conditions (as specified in Section 3.3), $\Pi_2^{P.O.}$ is safe from smart contract extortions.

3.1 Extortion threats siphoning collateral

Here we define a specific extortion threat in the setting of two-player games with deposit schemes where at least one player (the extortioner) has control over the release of the deposit of the other player (the victim). The extortion threat comes in the form of the extortioner threatening to burn the victim's deposit unless the victim transfers some amount of funds to the extortioner. Let $f_v > 0$ denote the victim's funds and d_v denote the victim's deposit. Let $f_e > 0$ denote the extortioner's funds. We can formally define the extortion threat as a two-player, 3 round extensive form game corresponding to the game tree in Figure 5 in Appendix C:

Two-player extortion game Γ_{extort}:
- The extortioner plays $x \in [0, \min(f_v, d_v)]$ which denotes the sum (with solvency restrictions) that the extortioner demands from the victim.
- The victim plays a bit $b_v \in \{0, 1\}$ which denotes whether the victim will yield to the threat or not. That is, the victim pays x to the extortioner if $b_v = 1$ and nothing otherwise.
- In the final round of Γ_{extort}, the extortioner plays $b_d \in \{0, 1\}$ which denotes whether the extortioner will burn the victim's deposit. If $b_d = 1$, the extortioner burns the deposit d_v. Otherwise the extortioner releases the deposit.

We first observe that naive extortion threats, i.e., extortion threats without using smart contracts, are non-effective in this setting without a trusted third party. Intuitively, this is because naive extortion threats are not binding: even if the victim plays $b_v = 1$ and gives x to the extortioner, the extortioner might still play $b_d = 1$ at the next round and burn the victim's deposit, leading to a loss of $d_v + x$ for the victim. The proof of the lemma is in Appendix F.1.

Lemma 3. *Naive extortion threats are non-effective without a trusted third party.*

3.2 Effective extortion threats via smart contracts

The above issue with naive extortion threats being non-effective is easily resolved with smart contracts. In contrast to naive extortion threats, smart contracts can be used to make a *public, binding commitment* to any strategy. The basic idea is for the extortioner to use auxiliary smart contracts to lock down a larger sum of money y_e compared to the extortion amount x. The auxiliary contract should also stipulate that the sum of money can only be released to the extortioner if the extortioner plays according to some strategy σ. The extortioner then sends this auxiliary smart contract to the victim together with an explanation of the contract which informs the victim that playing any other strategy will result in

the extortioner losing y_e. In particular, in the case where the victim yields to the threat and the extortioner burns the deposit, the payoff of the extortioner is $f_e - y_e + x < f_e$ which is even lower than the case where the extortioner does not extort the victim. Since the victim knows the extortioner is rational, the victim will also know that the extortioner will definitely play according to σ. The extortion threat is binding and hence effective.

Counter extortion attack. An obvious drawback of smart contract extortion attacks is that the public, binding nature of smart contracts, together with strong identities, makes smart contracts extortions undeniable. If we assume the existence of a working judicial system, an extortion threat using smart contracts would produce sufficient digital evidence to link the extortioner to the threat and can be potentially used by the victim in the judicial system to reclaim their funds. We provide more details on the undeniable property of smart contract extortions in Appendix C.2. We observe a further drawback of smart contract extortion threats: the extortioner can potentially be counter extorted by the victim. Specifically, the victim can exploit the fact that the extortioner's locked funds can only be released when the victim decides to yield or not. The victim can then make a counter demand for $x' \in [x, y_e]$ where y_e is the sum the extortioner locked inside their smart contract. To make the counter extortion threat effective, the victim can similarly use another smart contract to lock $y_v > x'$ such that the victim will lose y_v unless they play according to a fixed strategy.

Formally, we model a counter extortion threat by simply extending the basic single extortion step game Γ_{extort} to a 2 step extortion game that allows the victim to respond to the extortioner with a sum the victim wishes to counter extort. In fact, one can generalise the extortion game Γ_{extort} to any $n > 0$ step extortion game (i.e., n overall extortion moves by both extortioner and victim).

An interesting question is how long can the extortion game with counter extortions go on, i.e., how big can n be. In the totally unrealistic scenario where the extortioner and victim both have unlimited funds, n is infinite and the extortion game never terminates. However, if we adopt a set of more realistic assumptions, for instance finite amounts of funds, the existence of an external judicial system that forces the extortioner to return the victim's deposit, imposes a penalty of F_p on the extortioner, and finally costs F_l in terms of legal fees for the victim, we show that the game actually terminates with interesting consequences under some assumptions on these parameters.

Let us denote by $s_e(n) := \sum_{i=1}^{\lceil n/2 \rceil} y_e^i$ the sum of funds locked by the extortioner in all auxiliary smart contracts at the nth step of Γ_{extort}. We first show (with proof in Appendix F.2) that if the extortioner has funds at most that of the victim (i.e., $f_e \leq f_v$) the n step extortion game terminates at the first n that satisfies $s_e(n) \geq f_e$. In particular, we note that if the victim plays tit-for-tat, the game terminates with the victim successfully counter extorting the extortioner.

Lemma 4. *Suppose $f_e \leq f_v$. Let $n > 0$ be the first n that satisfies $s_e(n) \geq f_e$. The game terminates at step n with the victim counter extorting the extortioner.*

An important implication of Lemma 4 in the case where the extortioner has at most the amount of funds of the victim is that any smart contract extortion attack will never be successful. This situation is most promising for the victim as the victim's payoff after successfully counter extorting the extortioner is f_v, that is, the victim does not lose out on anything in the process of getting back their deposit. In Appendix C.3, we analyse two less promising cases for the victim where the extortioner has more funds than the victim. The former is the most pessimistic case where legal fees are higher than in the extortion amount and we show that the victim yields immediately. In the latter, the legal fees are smaller and the victim yields immediately but pays potentially less than the former case.

3.3 Extortion threats in $\Pi_2^{P.O.}$

In $\Pi_2^{P.O.}$, both players interact with a smart contract as well as handle the return of the collateral of the other party. Thus, we naturally have to consider the setting where players can employ auxiliary smart contracts to make extortion threats as described in the extortion game Γ_{extort} in Section 3.1.

In this section, we analyse our protocol $\Pi_2^{P.O.}$ in the one-shot setting in the context of extortion attacks with the following assumptions:

1. Both players start off with initial funds amounting to their deposit amounts d_i, and cannot access additional funds apart from what is paid out from the underlying game throughout the duration of $\Pi_2^{P.O.}$.
2. Players have access to an external judicial system with legal fees F_l.

Let $\mathfrak{A}$ denote Pareto optimal action profiles that give both players the same utility, i.e., $\mathfrak{A} := a \in \bar{A} \mid u_0(a) = u_1(a)$. We have the following propositions:

Proposition 1. *If we additionally restrict the support of μ to $\mathfrak{A}$, then $\Pi_2^{P.O.}$ is safe from smart contract extortions.*

Proposition 2. *If there are action profiles in* $\mathrm{supp}(\mu)$ *that do not give both players the same utility, then $\Pi_2^{P.O.}$ is unsafe from smart contract extortions.*

The proofs of Propositions 1 and 2 are simple applications of Lemmas 4, 5 and 6 and we defer these proofs to Appendices F.3 and F.4 respectively. Proposition 1 implies that if we restrict μ to only sample the action profile (C, C) in Chicken like in the case of Section 2.3, both players are safe from smart contract extortions as both players always receive the same utility from the sampled action profile. Although players are not safe from smart contract extortion threats in the setting where μ might sample action profiles that give unequal utility, we note that the maximum extortion amount is dependent on the legal fees. In particular, if we assume a setting where legal fees are negligible, the cost of being a victim of extortion is also negligible.

Finally, we stress that the assumption that players start off only with their deposit amounts as initial funds is not an unreasonable assumption: our work

assumes players are in the one-shot setting, and in the one-shot setting, the assumption that players do not bring additional funds apart from the deposit is the simplest possible assumption which renders the analysis in an already open-ended setting tractable.

4 On removing the simplifying assumptions in practice

As stated in Section 2.2, our analysis of protocol $\Pi_2^{P.O.}$ relies on a few simplifying assumptions. In this section, we address how these simplifying assumptions can be lifted in a practical implementation of our protocol.

The first assumption is the trusted sampling of the action profile from μ. To keep our protocol simple and modular, we propose using readily available public randomness beacons to obtain the random seed, for instance the NIST beacon [33]. We describe how to do this in more detail in Appendix G, as well as go into the shortcomings of existing and more complicated approaches.

The second assumption is the existence of simultaneous broadcast channels for players to simultaneously send their confirmations to the smart contract. The confirmations have to be simultaneous as the player that confirms first reveals their confirmation message to the other player and is disadvantaged. However, simultaneous broadcast channels are notoriously problematic to implement [35].

In Appendix H.1, we propose a procedure that uses time-lock puzzles and symmetric key encryption to replace a simultaneous broadcast channel. The basic idea is that messages need not be sent at the same time if the protocol can ensure that both messages reach the smart contract after time t_2 and that no player can "see" the other player's message before time t_2, and we use time-lock puzzles (refer to Appendix D.1) to achieve this.

Let $\tilde{\Pi}_2^{P.O.}$ be the exact same protocol as $\Pi_2^{P.O.}$ with the exception of using the time-lock encryption and decryption protocol instead of simultaneous broadcast channels. In the following, we show (with proof in Appendix H.2) a computational variant of Theorem 1 for the game $\tilde{\Gamma}'$ induced by $\tilde{\Pi}_2^{P.O.}$. The definitions of the computational variant of subgame perfect equilibrium and renegotiation-safety are given in Definition 5 in Appendix A.1 and Appendix H.2 respectively.

Theorem 2. *If we impose the same restrictions on μ as in Theorem 1, the strategy $\nu = (a, (b = 1, b = 1))$ is a computational renegotiation-safe subgame perfect equilibrium in the two-player, two-round extensive form game $\tilde{\Gamma}'$ induced by $\tilde{\Pi}_2^{P.O.}$.*

Acknowledgments

Pavel Hubáček was partially supported by the Academy of Sciences of the Czech Republic (RVO 67985840), Czech Science Foundation GAČR grant No. 25-16311S, and by the ERC-CZ project LL2406 of the Ministry of Education, Youth and Sports of the Czech Republic. Michelle Yeo is supported by the MOE START-NTU overseas postdoctoral fellowship award.

References

1. Ackermann, L., Arapinis, M., Georgiou, P., Lamprou, N., Mareková, L., Zacharias, T.: E-cclesia: Universally composable self-tallying elections over anonymous broadcast. IACR Commun. Cryptol. **2**(3), 8 (2025)
2. Alvarez, M.M., Arneson, H., Berger, B., Bousfield, L., Buckland, C., Edelman, Y., Felten, E.W., Goldman, D., Jordan, R., Kelkar, M., Mamageishvili, A., Ng, H., Sanghi, A., Shoup, V., Tsao, T.: Bold: Fast and cheap dispute resolution. In: AFT. LIPIcs, vol. 316, pp. 2:1–2:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2024)
3. Andrychowicz, M., Dziembowski, S., Malinowski, D., Mazurek, L.: Fair two-party computations via bitcoin deposits. In: Böhme, R., Brenner, M., Moore, T., Smith, M. (eds.) Financial Cryptography and Data Security - FC 2014 Workshops, BITCOIN and WAHC 2014, Christ Church, Barbados, March 7, 2014, Revised Selected Papers. Lecture Notes in Computer Science, vol. 8438, pp. 105–121. Springer (2014). https://doi.org/10.1007/978-3-662-44774-1_8, https://doi.org/10.1007/978-3-662-44774-1_8
4. Arapinis, M., Kocsis, Á., Lamprou, N., Medley, L., Zacharias, T.: Universally composable simultaneous broadcast against a dishonest majority and applications. CoRR **abs/2305.06468** (2023)
5. Ashlagi, I., Monderer, D., Tennenholtz, M.: On the value of correlation. CoRR **abs/1207.1362** (2012), http://arxiv.org/abs/1207.1362
6. Atallah, M.J., Blanton, M., Frikken, K.B., Li, J.: Efficient correlated action selection. In: Crescenzo, G.D., Rubin, A.D. (eds.) Financial Cryptography and Data Security, 10th International Conference, FC 2006, Anguilla, British West Indies, February 27-March 2, 2006, Revised Selected Papers. Lecture Notes in Computer Science, vol. 4107, pp. 296–310. Springer (2006). https://doi.org/10.1007/11889663_24, https://doi.org/10.1007/11889663_24
7. Aumann, R.J.: Subjectivity and correlation in randomized strategies. Journal of Mathematical Economics **1**(1), 67–96 (1974). https://doi.org/https://doi.org/10.1016/0304-4068(74)90037-8, https://www.sciencedirect.com/science/article/pii/0304406874900378
8. Aumann, R.J., Hart, S.: Long Cheap Talk. Econometrica **71**(6), 1619–1660 (November 2003), https://ideas.repec.org/a/ecm/emetrp/v71y2003i6p1619-1660.html
9. Aumann, Y., Lindell, Y.: Security against covert adversaries: Efficient protocols for realistic adversaries. J. Cryptol. **23**(2), 281–343 (2010). https://doi.org/10.1007/s00145-009-9040-7, https://doi.org/10.1007/s00145-009-9040-7
10. Barakbayeva, T., Cai, Z., Goharshady, A.K., Keypoor, K.: Smart contracts for trustless sampling of correlated equilibria. In: IJCAI. pp. 3744–3752. ijcai.org (2025)
11. Bárány, I.: Fair distribution protocols or how the players replace fortune. Math. Oper. Res. **17**(2), 327–340 (1992). https://doi.org/10.1287/moor.17.2.327, https://doi.org/10.1287/moor.17.2.327
12. Barman, S., Ligett, K.: Finding any nontrivial coarse correlated equilibrium is hard. In: Roughgarden, T., Feldman, M., Schwarz, M. (eds.) Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015. pp. 815–816. ACM (2015). https://doi.org/10.1145/2764468.2764497, https://doi.org/10.1145/2764468.2764497

13. Bentov, I., Kumaresan, R.: How to use bitcoin to design fair protocols. In: Garay, J.A., Gennaro, R. (eds.) Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II. Lecture Notes in Computer Science, vol. 8617, pp. 421–439. Springer (2014). https://doi.org/10.1007/978-3-662-44381-1_24, https://doi.org/10.1007/978-3-662-44381-1_24
14. Bentov, I., Kumaresan, R., Miller, A.: Instantaneous decentralized poker. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10625, pp. 410–440. Springer (2017). https://doi.org/10.1007/978-3-319-70697-9_15, https://doi.org/10.1007/978-3-319-70697-9_15
15. Chatterjee, K., Gilbert, S., Schmid, S., Svoboda, J., Yeo, M.: When is liquid democracy possible?: On the manipulation of variance. In: PODC. pp. 241–251. ACM (2025)
16. Chatterjee, K., Goharshady, A.K., Pourdamghani, A.: Probabilistic smart contracts: Secure randomness on the blockchain. In: IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2019, Seoul, Korea (South), May 14-17, 2019. pp. 403–412. IEEE (2019). https://doi.org/10.1109/BLOC.2019.8751326, https://doi.org/10.1109/BLOC.2019.8751326
17. Damgård, I., Keller, M., Larraia, E., Pastro, V., Scholl, P., Smart, N.P.: Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In: Crampton, J., Jajodia, S., Mayes, K. (eds.) Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings. Lecture Notes in Computer Science, vol. 8134, pp. 1–18. Springer (2013). https://doi.org/10.1007/978-3-642-40203-6_1, https://doi.org/10.1007/978-3-642-40203-6_1
18. Daskalakis, C., Goldberg, P.W., Papadimitriou, C.H.: The complexity of computing a Nash equilibrium. SIAM J. Comput. **39**(1), 195–259 (2009). https://doi.org/10.1137/070699652, https://doi.org/10.1137/070699652
19. Dodis, Y., Halevi, S., Rabin, T.: A cryptographic solution to a game theoretic problem. In: Bellare, M. (ed.) Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1880, pp. 112–130. Springer (2000). https://doi.org/10.1007/3-540-44598-6_7, https://doi.org/10.1007/3-540-44598-6_7
20. Farrell, J., Rabin, M.: Cheap talk. Journal of Economic Perspectives **10**(3), 103–118 (September 1996). https://doi.org/10.1257/jep.10.3.103, https://www.aeaweb.org/articles?id=10.1257/jep.10.3.103
21. Faust, S., Hazay, C., Kretzler, D., Schlosser, B.: Financially backed covert security. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13178, pp. 99–129. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_4, https://doi.org/10.1007/978-3-030-97131-1_4
22. Goharshady, A.K.: Irrationality, extortion, or trusted third-parties: Why it is impossible to buy and sell physical goods securely on the blockchain. In: Xiang, Y., Wang, Z., Wang, H., Niemi, V. (eds.) 2021 IEEE International Conference

on Blockchain, Blockchain 2021, Melbourne, Australia, December 6-8, 2021. pp. 73–81. IEEE (2021). https://doi.org/10.1109/Blockchain53845.2021.00021, https://doi.org/10.1109/Blockchain53845.2021.00021
23. Goyal, V., Mohassel, P., Smith, A.D.: Efficient two party and multi party computation against covert adversaries. In: Smart, N.P. (ed.) Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings. Lecture Notes in Computer Science, vol. 4965, pp. 289–306. Springer (2008). https://doi.org/10.1007/978-3-540-78967-3_17, https://doi.org/10.1007/978-3-540-78967-3_17
24. Gradwohl, R., Livne, N., Rosen, A.: Sequential rationality in cryptographic protocols. ACM Trans. Economics and Comput. **1**(1), 2:1–2:38 (2013). https://doi.org/10.1145/2399187.2399189, https://doi.org/10.1145/2399187.2399189
25. Hu, Q., Nigam, Y., Wang, Z., Wang, Y., Xiao, Y.: A correlated equilibrium based transaction pricing mechanism in blockchain. In: IEEE ICBC. pp. 1–7. IEEE (2020)
26. Hubáček, P., Nielsen, J.B., Rosen, A.: Limits on the power of cryptographic cheap talk. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 277–297. Springer (2013). https://doi.org/10.1007/978-3-642-40041-4_16, https://doi.org/10.1007/978-3-642-40041-4_16
27. Hubáček, P., Park, S.: Cryptographically blinded games: leveraging players' limitations for equilibria and profit. In: Babaioff, M., Conitzer, V., Easley, D.A. (eds.) ACM Conference on Economics and Computation, EC '14, Stanford , CA, USA, June 8-12, 2014. pp. 207–208. ACM (2014). https://doi.org/10.1145/2600057.2602903, https://doi.org/10.1145/2600057.2602903
28. Izmalkov, S., Lepinski, M., Micali, S.: Verifiably secure devices. In: Canetti, R. (ed.) Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008. Lecture Notes in Computer Science, vol. 4948, pp. 273–301. Springer (2008). https://doi.org/10.1007/978-3-540-78524-8_16, https://doi.org/10.1007/978-3-540-78524-8_16
29. Izmalkov, S., Lepinski, M., Micali, S.: Perfect implementation. Games Econ. Behav. **71**(1), 121–140 (2011). https://doi.org/10.1016/j.geb.2010.05.003, https://doi.org/10.1016/j.geb.2010.05.003
30. Jarrah, M.H.A., Zhang, J.: Trusted mediator agents to better manage complex and competitive supply chains. In: Weiss, G., Yolum, P., Bordini, R.H., Elkind, E. (eds.) Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015. pp. 1811–1812. ACM (2015), http://dl.acm.org/citation.cfm?id=2773449
31. Kahng, A., Mackenzie, S., Procaccia, A.D.: Liquid democracy: An algorithmic perspective. In: AAAI. pp. 1095–1102. AAAI Press (2018)
32. Kalodner, H.A., Goldfeder, S., Chen, X., Weinberg, S.M., Felten, E.W.: Arbitrum: Scalable, private smart contracts. In: USENIX Security Symposium. pp. 1353–1370. USENIX Association (2018)
33. Kelsey, J., Brandão, L.T.A.N., Peralta, R., Booth, H.: Nistir 8213. a reference for randomness beacons: Format and protocol version 2. Tech. rep. (2011)
34. Kiayias, A., Yung, M.: Self-tallying elections and perfect ballot secrecy. In: Public Key Cryptography. Lecture Notes in Computer Science, vol. 2274, pp. 141–158. Springer (2002)

35. Kol, G., Naor, M.: Games for exchanging information. In: Dwork, C. (ed.) Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008. pp. 423–432. ACM (2008). https://doi.org/10.1145/1374376.1374437, https://doi.org/10.1145/1374376.1374437
36. Kumaresan, R., Bentov, I.: How to use bitcoin to incentivize correct computations. In: Ahn, G., Yung, M., Li, N. (eds.) Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014. pp. 30–41. ACM (2014). https://doi.org/10.1145/2660267.2660380, https://doi.org/10.1145/2660267.2660380
37. Lepinski, M., Micali, S., Shelat, A.: Collusion-free protocols. In: Gabow, H.N., Fagin, R. (eds.) Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005. pp. 543–552. ACM (2005). https://doi.org/10.1145/1060590.1060671, https://doi.org/10.1145/1060590.1060671
38. Mailath, G.J., Samuelson, L.: Repeated Games and Reputations: Long-Run Relationships. Oxford University Press (2006)
39. Moulin, H., Vial, J.P.: Strategically zero-sum games: the class of games whose completely mixed equilibria cannot be improved upon. International Journal of Game Theory **7**, 201–221 (1978)
40. Osborne, M.J., Rubinstein, A.: A course in game theory. The MIT Press, Cambridge, USA (1994), electronic edition
41. Papadimitriou, C.H., Roughgarden, T.: Computing correlated equilibria in multi-player games. J. ACM **55**(3), 14:1–14:29 (2008). https://doi.org/10.1145/1379759.1379762, https://doi.org/10.1145/1379759.1379762
42. Pass, R., Shelat, A.: Renegotiation-safe protocols. In: Chazelle, B. (ed.) Innovations in Computer Science - ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings. pp. 61–78. Tsinghua University Press (2011), http://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/26.html
43. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Tech. rep. (1996)
44. Schwartzbach, N.I.: Deposit schemes for incentivizing behavior in finite games of perfect information. CoRR **abs/2107.08748** (2021), https://arxiv.org/abs/2107.08748
45. Schwartzbach, N.I.: Payment schemes from limited information with applications in distributed computing. In: Pennock, D.M., Segal, I., Seuken, S. (eds.) EC '22: The 23rd ACM Conference on Economics and Computation, Boulder, CO, USA, July 11 - 15, 2022. pp. 129–149. ACM (2022). https://doi.org/10.1145/3490486.3538342, https://doi.org/10.1145/3490486.3538342
46. Sen, S., Sajja, N.: Robustness of reputation-based trust: boolean case. In: The First International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2002, July 15-19, 2002, Bologna, Italy, Proceedings. pp. 288–293. ACM (2002). https://doi.org/10.1145/544741.544808, https://doi.org/10.1145/544741.544808
47. Seres, I.A., Glaeser, N., Bonneau, J.: Short paper: Naysayer proofs. In: FC (2). Lecture Notes in Computer Science, vol. 14745, pp. 22–32. Springer (2024)
48. Teague, V.: Selecting correlated random actions. In: Juels, A. (ed.) Financial Cryptography, 8th International Conference, FC 2004, Key West, FL, USA, February 9-12, 2004. Revised Papers. Lecture Notes in Computer Science, vol. 3110, pp.

181–195. Springer (2004). https://doi.org/10.1007/978-3-540-27809-2_20, https://doi.org/10.1007/978-3-540-27809-2_20
49. Teutsch, J., Reitwießner, C.: A scalable verification solution for blockchains. CoRR **abs/1908.04756** (2019)
50. Yu, B., Singh, M.P.: An evidential model of distributed reputation management. In: The First International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2002, July 15-19, 2002, Bologna, Italy, Proceedings. pp. 294–301. ACM (2002). https://doi.org/10.1145/544741.544809, https://doi.org/10.1145/544741.544809
51. Zhang, H., Merino, L., Qu, Z., Bastankhah, M., Estrada-Galiñanes, V., Ford, B.: F3B: A low-overhead blockchain architecture with per-transaction front-running protection. In: AFT. LIPIcs, vol. 282, pp. 3:1–3:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2023)
52. Zhang, H., Yeo, M., Estrada-Galiñanes, V., Ford, B.: Zeroauction: Zero-deposit sealed-bid auction via delayed execution. In: FC Workshops. Lecture Notes in Computer Science, vol. 14746, pp. 170–188. Springer (2024)
53. Zhu, R., Ding, C., Huang, Y.: Efficient publicly verifiable 2PC over a blockchain with applications to financially-secure computations. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019. pp. 633–650. ACM (2019). https://doi.org/10.1145/3319535.3363215, https://doi.org/10.1145/3319535.3363215
54. Zimbeck, D., Donato, S., Hahn, A., Sloin, P., Meacci, G.: Bithalo: Mother of smart contracts and a decentralized market for everything. Tech. rep. (2014)

A Game theoretic notions

A.1 Games and basic solution concepts

Let $\Gamma = (N, (A_i), (u_i))$ be an N player game where A_i is a finite set of actions for each player $i \in [N]$ and denote by $A := A_1 \times \cdots \times A_N$ the set of action profiles. The utility function of each player i, $u_i : A \to \mathbb{R}$, gives the payoff player i gets when an action profile $a \in A$ is played. A *strategy* $\sigma_i \in \Delta(A_i)$ of a player $i \in [N]$ is a distribution over all possible actions of the player.

Definition 3. *(Nash Equilibrium). A Nash Equilibrium (NE) of Γ is a product distribution $\alpha \in \times_{j\in[N]}\Delta(A_j)$ such that for every player $i \in [N]$ and for all a'_i in A_i,*

$$\mathbb{E}_{a\leftarrow\alpha}[u_i(a)] \geq \mathbb{E}_{a\leftarrow\alpha}[u_i(a'_i, a_{-i})]$$

In some games, players can achieve better payoffs if they correlate their strategies. This is typically done using some trusted correlation device called a mediator that samples an action profile a from some distribution in $\Delta(A)$ and gives a_i *privately* as a signal to player i. The distribution the mediator samples from is called a correlated equilibrium if, after seeing the signal, no player has any incentive to deviate.

Definition 4. *(Correlated Equilibrium). A Correlated Equilibrium (CE) of Γ is a distribution $\alpha \in \Delta(A)$ such that for every player $i \in [N]$ and for all b_i, a'_i in A_i satisfying $\mathbb{P}_{a\leftarrow\alpha}[a_i = b_i] > 0$,*

$$\mathbb{E}_{a\leftarrow\alpha}[u_i(a) \mid a_i = b_i] \geq \mathbb{E}_{a\leftarrow\alpha}[u_i(a'_i, a_{-i}) \mid a_i = b_i]$$

The set of correlated equilibria is a superset of the set of Nash Equilibria as sampling action profiles from a product distribution does not give any player additional information about the actions of other players when conditioning on the advice from the mediator.

Example 1. In the game of chicken with payoffs depicted in Table 1, the distribution that selects action profiles $(C, C), (C, D)$ and (D, C) with probability $\frac{1}{3}$ is a correlated equilibrium. The expected payoff of this correlated equilibrium for each player is $5 > \frac{14}{3}$ which is the expected payoff of the mixed strategy Nash equilibrium of the game of chicken.

Thus far, we have only defined solution concepts in single-shot games. Multi-round games where players' actions arrive sequentially are modelled as extensive-form games. For a formal definition of extensive-form games, see, e.g., [40]. For our purposes, however, one can simply think of extensive-form games as defined by a game tree T. All non-leaf vertices in T are partitioned into sets with each set corresponding to one player in the game. A player move (or action) at vertex x in their vertex set is simply a choice of an edge from x to some child of x. A path from the root of T to a leaf vertex corresponds to a sequence of player moves made by the players in the game. Each leaf of T is labelled with a utility vector

which shows how much utility each player gets when the game play terminates at this leaf. In the imperfect information or simultaneous action settings, the vertices belonging to any player are further partitioned into information sets I which capture the idea that a player making a move at any vertex $x \in I$ is uncertain whether they are making the move from x or any other vertex $x' \in I$.

A subgame of an extensive-form game corresponds to a subtree in T rooted at any non-leaf vertex x that belongs to its own information set I, i.e., there are no other vertices that are in I except for x. A strategy profile is a *subgame perfect equilibrium* if it is a Nash equilibrium for all subgames in the game.

Finally, we also consider the setting where players are computationally bounded, i.e., restricted to PPT. In this setting, the negligible probability that players might break the underlying cryptographic primitives could result in the players achieving a higher utility than when behaving honestly. A variant of Nash Equilibrium which is better suited for this setting is the computational Nash Equilibrium, which accounts for this negligible probability of players breaking the underlying cryptographic primitives [19].

Definition 5. *(Computational Nash Equilibrium). A Computational Nash Equilibrium of Γ is a product distribution $\alpha \in \times_{j\in[N]}\Delta(A_j)$ such that for every player $i \in [N]$,*

1. *α_i is computable in PPT*
2. *for all a'_i in A_i, $\mathbb{E}_{a\leftarrow\alpha}[u_i(a)]+\mathsf{negl}(n)+\mathsf{negl}(k_0)+\mathsf{negl}(k_1) \geq \mathbb{E}_{a\leftarrow\alpha}[u_i(a'_i, a_{-i})]$*

A.2 Pareto optimality

For any player $i \in [N]$ and action profiles $a, a' \in A$, we say *a dominates a' for player i* if $u_i(a) > u_i(a')$. We say an action profile a is *Pareto optimal* if $\nexists a' \in A$, $a' \neq a$, such that $\exists i$ where $u_i(a') > u_i(a)$ and for all $j \neq i, u_j(a') \geq u_j(a)$. That is, no other action profile can strictly improve at least one player's utility while keeping the utility of the other players at least the same. For any normal-form game $\Gamma = (N, (A_i), (u_i))$, we denote by $\bar{A}$ the set of all Pareto optimal action profiles in Γ.

The notion of Pareto optimality extends to distributions over action profiles in a natural way. We say a distribution $\mu \in \Delta(A)$ over action profiles *dominates* distribution $\mu' \in \Delta(A)$ if the expected utility of all players under μ is at least as large as that under μ', i.e., for all $i \in [N]$, it holds that $\mathbb{E}_{a\leftarrow\mu}[u_i(a)] \geq \mathbb{E}_{a\leftarrow\mu'}[u_i(a)]$, and there is at least one player that gets strictly greater expected utility under μ, i.e., there exists $i \in [N]$ such that $\mathbb{E}_{a\leftarrow\mu}[u_i(a)] > \mathbb{E}_{a\leftarrow\mu'}[u_i(a)]$. Given a set $\mathcal{D} \subseteq \Delta(A)$ of distributions of action profiles, we say a distribution $\mu \in \mathcal{D}$ is *Pareto optimal in $\mathcal{D}$* if no distribution $\mu' \in \mathcal{D}$ dominates μ. The above definition is commonly used to restrict the sets of Nash equilibria (respectively correlated equilibria), giving rise to *Pareto optimal Nash equilibria* (respectively *Pareto optimal correlated equilibria*). In this work, we also consider the Pareto optimal distributions in $\Delta(\bar{A})$, i.e., the Pareto optimal distributions supported on Pareto optimal action profiles.

Inputs and parameters:
- a two-player normal-form game $\Gamma = (A_1, A_2, u_1, u_2)$
- distribution $\mu \in \Delta(A_1 \times A_2)$
- for $i \in \{0, 1\}$, deposit amount $d_i = \max_{a \in A}(u_i(a)) - \min_{a \in \mathrm{supp}(\mu)}(u_i(a)) + 1$

Setup phase:
1. P_0 and P_1 deposit d_0 and d_1 coins into $\mathcal{SC}$ as collateral.
2. P_0 and P_1 agree on three times:
   - time t_0 when $\mathcal{SC}$ would sample an action profile $a \leftarrow \mu$,
   - time t_1 when they would play the game Γ,
   - time t_2 where they would report the action the other player played to $\mathcal{SC}$.

Game phase:
1. At t_0, $\mathcal{SC}$ samples $a \leftarrow \mu$ and publishes a to P_0 and P_1.
2. At t_1, both players simultaneously play their actions in the underlying game Γ. We denote by $\tilde{a}$ the action profile played by the players.
3. At t_2:
   - P_0 and P_1 simultaneously send confirmation bits b_0 and b_1 to $\mathcal{SC}$.
   - Given b_0 and b_1, $\mathcal{SC}$ resolves the collateral deposits as follows:
     If both players input $b_0 = b_1 = 1$, $\mathcal{SC}$ returns d_0 and d_1 to P_0 and P_1 respectively.
     If both players input $b_0 = b_1 = 0$, $\mathcal{SC}$ burns both deposits.
     If P_i inputs $b_i = 1$ and P_{1-i} inputs $b_{1-i} = 0$, $\mathcal{SC}$ returns d_{1-i} to P_{1-i} and burns P_i's deposit.

Fig. 3: Our protocol.

B Protocol details

B.1 Protocol description

B.2 Renegotiation-safety details

Here we provide more details about our notion of renegotiation-safety and how it differs from the original notion proposed by Pass and Shelat [42]. Renegotiation-safety as a stability concept mandates that no player can benefit from suggesting a different joint strategy that will not be prone to future renegotiations. Pass and Shelat gave a general definition of renegotiation-safety in extensive-form games. Our variant of renegotiation-safety is tailored to the games of interest for us, i.e., extensive-form games where players make simultaneous moves at each round.

The first item in Definition 2 differs from the original definition in [42]. It captures the condition that, in games with simultaneous moves in each round, a successful profitable renegotiation must not lower the utility of the opponents – this property implicitly rules out infinite cycles of renegotiations *in the same round* corresponding to competing counter-renegotiations that benefit only the proposing player while making the opponents worse off.

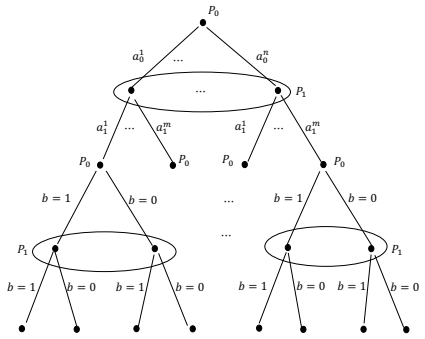

Fig. 4: Game tree corresponding to Γ'. The circled vertices in the game tree are vertices that belong to the same information set, i.e., there are effectively two rounds with simultaneous moves by the players.

B.3 Discussion details

One-shot vs. repeated setting. Although all action profiles are possible Nash equilibria of the subgames at round 2 of Γ', we argue that with an additional (but very reasonable) assumption, players actually prefer to pick the action profiles that correspond to how the other player played (i.e., the action profiles we selected in our proof of Lemma 1). We can assume the existence of some overarching reputation system (which is typical in multi-agent systems [46,50,38]) that players are part of. If either player picks an action that does not correspond to how the other player actually played, this would lower their reputation and consequently limit future interaction. The study of reputation in the repeated setting synergises particularly well with the blockchain setting as, firstly, in permissioned blockchains it is costly to launch Sybil attacks to tarnish someone's reputation. Additionally, for proof-of-stake based blockchains with the players being validators, deposit slashing can be combined with stake slashing to further cripple the power of cheating players. We note that the precise amount of stake to slash depends on the game itself as well as the equilibrium behaviour of the players in the repeated setting, and we thus leave this analysis to future work.

Extending our protocol to multiple players. In our 2 player protocol $\Pi_2^{P.O.}$, each player confirms the actions of the other player in the second round and the deposits are returned or slashed accordingly. Extending our current approach to the n-player setting is not completely trivial. A naive way to do so would simply be to let all n players confirm the actions of the other players. This, however, does not take into account potential coalitions that can be formed between groups of players, which could impact the stability of the protocol. Moreover, designing the optimal deposit scheme in this setting would have to involve analysing all possible player coalitions and account for the maximum difference in payoffs over these possible coalitions. Given that analysing the two-player case already

highlights various interesting aspects of the interaction of smart contracts and game-theory, we leave the extension to n players for future research.

C Analysis of extortions details

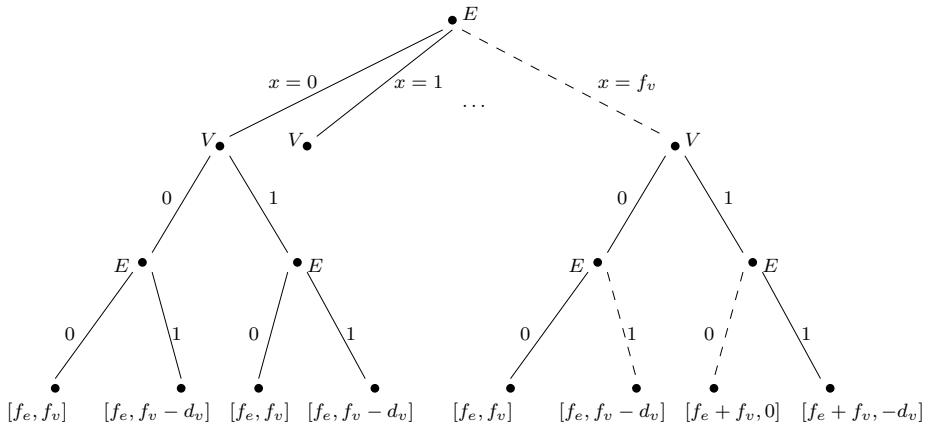

Fig. 5: Game tree corresponding to Γ_{extort}. for the case where the victim has less funds than the current deposit so the maximum amount the extortioner can extort is f_v. Vertices labelled E (resp. V) are extortioner (resp. victim) vertices. The first (resp. second) element of the utility vector at each leaf is the utility of extortioner (resp. victim). The dashed edges correspond to a committed strategy of the extortioner.

C.1 Effective extortion threats using auxiliary smart contracts.

Formally, we say an extortioner uses a smart contract to make a threat at a given extortioner vertex v in the game tree when the extortioner selects a set of paths from v to the leaves of the tree and commits to them. The paths have to satisfy the additional consistency constraint that for any two paths p_1, p_2 in this set with the same move history of the victim at any extortioner vertex $v \in p_1 \cap p_2$, the edge beginning at v has to be the same in p_1 and p_2. A move history of a player on a given path p at a vertex v is simply the sequence of edges that begins with the player's vertex in the prefix of v in p (i.e., the subpath of p from the root to v). Note that all such prefixes are unique in a tree. The above constraint is necessary to ensure that the extortioner cannot commit to inconsistent strategies, for instance in the tree depicted in Figure 5 the extortioner cannot commit to both releasing the deposit and burning it on all paths where the victim yields to the threat.

C.2 Undeniability of smart contract extortions

Although smart contracts can be employed to make extortion threats effective, they come at a cost of deniability. Deniability in this setting can happen in two ways: given a transcript of interactions (i.e., all public timestamped and signed messages) between the extortioner and a victim, the extortioner can either convince an external third party that the extortion threat did not happen, or that they did not create the threat. In the former case, the binding nature of the threat makes it impossible for the extortioner to claim they played a different strategy and thus the threat did not happen. In the latter case, if we assume strong identities, the public nature of the smart contract extortion threat makes it impossible to link the threat to some other identity given the transcript. As such, an extortion threat using smart contracts would produce sufficient digital evidence to link the extortioner to the threat and can be potentially used by the victim in any external judicial process.

C.3 Analysis of extortions when the extortioner has more funds

Here, we analyse the case where the extortioner has strictly more funds than the victim. Depending on legal fees, the situation for the victim is somewhat more pessimistic. We first analyse the worst case for the victim where the legal fees are higher than the extortion amount. Let ϵ be the smallest unit of funds and let $\delta = f_e - f_v$ be the difference between the extortioner and victim's funds.

Lemma 5. *If $F_l > x$ and $f_e > f_v$, the extortion game terminates after 1 step with the victim yielding to the extortion threat.*

Proof. If $F_l > x$, the utility of the victim when taking the extortioner to court is $f_v - F_l < f_v - x$ which is the utility of the victim when yielding to the threat. So a rational victim will always either yield to the threat immediately or attempt to counter extort the extortioner.

Now we show that any attempt of the victim to counter extort the extortioner will be unsuccessful. Consider the case where $\delta \geq 2\epsilon$. The extortioner starts the game with the maximum possible extortion threat of amount $\frac{f_e}{2} - \epsilon = \frac{f_v}{2} + \frac{\delta}{2} - \epsilon$ and locks $\frac{f_e}{2} + \epsilon = \frac{f_v}{2} + \frac{\delta}{2} + \epsilon$ into their auxiliary smart contract. Any counter extortion response of the victim needs to be some value $x' \in [\frac{f_v}{2} + \frac{\delta}{2} - \epsilon, \frac{f_v}{2} + \frac{\delta}{2} + \epsilon]$. This would mean the victim has funds amounting at least to $\frac{f_v}{2} + \frac{\delta}{2} - \epsilon + \frac{f_v}{2} + \frac{\delta}{2} = f_v + \delta - \epsilon > f_v$ which is a contradiction.

Consider the case where $\delta = \epsilon$. The extortioner starts the game with the threat of amount $\frac{f_v}{2}$ and locks $\frac{f_v}{2} + \epsilon$ into their auxiliary smart contract. As ϵ is the smallest unit of funds, any counter extortion response of the victim would require the victim having funds of at least $\frac{f_v}{2} + \frac{f_v}{2} + \epsilon = f_v + \epsilon > f_v$ which is a contradiction.

In the above case the payoff of the victim is $f_v - (\frac{f_v}{2} + \frac{\delta}{2} - \epsilon)$. Now we observe that if we reduce the legal fees, the situation gets better for the victim.

Lemma 6. *If $F_l < x$, $F_p > 0$, and $f_e > f_v$, the extortion game terminates after 1 step with the victim yielding to the extortion threat of amount $x = \min(F_l, \frac{f_v}{2} + \frac{\delta}{2} - \epsilon)$*

Proof. First we note that from Lemma 5, as long as the funds of the extortioner is strictly higher than that of the victim, the victim cannot successfully counter extort the extortioner and would therefore have to yield to the threat. All that remains in this situation is to analyse an upper bound on the amount the extortioner can extort from the victim.

We observe that if the extortioner attempts to extort $x > F_l$, the victim will go to court immediately and the extortioner would have to pay the fine of F_p which results in a utility of $f_e - F_p < f_e$. Thus the rational extortioner needs to ensure $x < F_l$.

From Lemma 5, the maximum amount the extortioner can extort from the victim is $\frac{f_v}{2} + \frac{\delta}{2} - \epsilon$ and together with the constraint that the extorted amount $x < F_l$, this gives an upper bound of $\min(F_l, \frac{f_v}{2} + \frac{\delta}{2} - \epsilon)$ on the extortion amount.

D Cryptographic definitions

D.1 Time-lock Puzzles

Time-lock puzzles were introduced in [43] as a solution to the problem of "sending information to the future", that is, to ensure some encrypted information cannot be decrypted until a certain amount of time has passed. The basic idea is to generate a puzzle that is intrinsically sequential and cannot be computed faster even when computation is parallelised. Thus, the solver of a time-lock puzzle with time parameter T would necessarily need to perform T sequential computation steps before getting the solution. The example used in [43] for constructing a time-lock puzzle consists of the puzzle giver selecting 3 elements: a modulus $n = pq$ where p and q are large primes, a random element $x \in \mathbb{Z}_n^*$, and finally T the number of sequential computation steps the puzzle solver would need to do. The puzzle giver then gives the tuple (n, x, T) to the puzzle solver. Importantly, $\phi(n)$, the order of the multiplicative group $\mathbb{Z}_n^*$, is not given to the puzzle solver.

To get the solution of the puzzle, the solver has to perform T repeated squarings, i.e.,

$$x \to x^2 \to x^{2^2} \to \cdots \to x^{2^T} \bmod n$$

The sequential nature of the computation stems from the conjecture that it is impossible to compute $x^{2^T} \bmod n$ with less than T steps when $\phi(n)$ is not known as this is as hard as factorising n.

For the puzzle giver, computing the solution (and hence verifying the correctness of the puzzle solution) is efficient as the puzzle giver knows $\phi(n) = (p-1)(q-1)$. The puzzle giver simply needs to first compute $e := 2^T \bmod \phi(n)$ and then compute solution $y = x^e \bmod n$.

D.2 Symmetric key encryption

We denote a negligible function in some parameter λ by $\mathsf{negl}(\lambda)$. We use PPT as a shorthand for probabilistic polynomial time.

Definition 6. *(Symmetric key encryption scheme) A symmetric key encryption scheme with security parameter 1^n is a tuple of 3 PPT algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ such that*

1. *The key generation algorithm takes as input the security parameter 1^n and outputs a key k. We denote this by $k \leftarrow \mathsf{Gen}(1^n)$*
2. *The encryption algorithm takes the key k as input and a message $m \in \{0,1\}^*$ and outputs a ciphertext c. We denote this by $c \leftarrow \mathsf{Enc}_k(m)$*
3. *The decryption algorithm takes the key k and ciphertext c as input and outputs a message m. This algorithm is deterministic and so we denote it as $m := \mathsf{Dec}_k(c)$.*

We require that the scheme is correct, i.e., for every n and every $k \leftarrow \mathsf{Gen}(1^n)$ and every $m \in \{0,1\}^*$, $\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = m$.

For our protocol, we require that the symmetric key encryption scheme is CPA secure against computationally bounded adversaries. CPA security is defined using the following CPA indistinguisability game for any adversary $\mathcal{A}$, challenger $\mathcal{C}$ any security parameter n and any symmetric key encryption scheme Pi:

CPA indistinguishability game $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$

1. $k \leftarrow \mathsf{Gen}(1^n)$.
2. $\mathcal{A}$ gets 1^n and oracle access to $\mathsf{Enc}_k(\cdot)$. $\mathcal{A}$ outputs a pair of messages m_0, m_1 of the same length and gives them to $\mathcal{C}$.
3. $\mathcal{C}$ chooses a random bit b and computes challenge ciphertext $c \leftarrow \mathsf{Enc}_k(m_b)$. c is given to $\mathcal{A}$.
4. $\mathcal{A}$ outputs a bit b'. The output of the game is 1 if $b' = b$ and 0 otherwise.

Definition 7. *(CPA-secure symmetric encryption scheme) A symmetric key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is CPA-secure if for all PPT adversaries $\mathcal{A}$ there exists a negligible function negl such that*

$$\mathbb{P}[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n)$$

E Analysis of our protocol

E.1 Proof of Lemma 1

Lemma 1. *Let $\Gamma = (A_1, A_2, u_1, u_2)$ be a normal-form game and $\mu \in \Delta(A)$ a distribution of action profiles. The strategy profile ν from Definition 1 is a subgame perfect equilibrium in Γ'.*

		P_1	
		a_1	$\neg a_1$
P_0	a_0	$(u_0(a), u_1(a))$	$(u_0(h), u_1(h) - d_1)$
	$\neg a_0$	$(u_0(h) - d_0, u_1(h))$	$(u_0(h) - d_0, u_1(h) - d_1)$

Table 2: Payoff matrix for the reduced game at round 1. All subgames at round 2 of Γ' are eliminated and replaced with the payoffs of a Nash equilibrium in that subgame. h denotes the history (i.e., action profile) played at round 1 and a denotes the sampled action. For each player i, $\neg a_i$ denotes the player choosing to deviate from the sampled action profile.

		P_1	
		$b = 1$	$b = 0$
P_0	$b = 1$	$(u_0(h), u_1(h))$	$(u_0(h) - d_0, u_1(h))$
	$b = 0$	$(u_0(h), u_1(h) - d_1)$	$(u_0(h) - d_0, u_1(h) - d_1)$

Table 3: Payoff matrix for all the subgames at round 2 of Γ'. h denotes the history (i.e., action profile) played at round 1.

Proof. We proceed by backwards induction. Recall that the payoff matrix in Table 3 defines the payoff corresponding to all possible action profiles for all the subgames at round two of Γ'. We observe that all these action profiles are possible Nash equilibria of these subgames as any unilateral deviation can only affect the deposit and hence payoff of the other player. As such, we can simply pick any equilibrium from each subgame. For all subgames corresponding to the history where both players deviated at round 1 of Γ' we pick $(b = 0, b = 0)$ as the equilibrium. For all subgames corresponding to the history where only 1 player deviated from the sampled action at round 1 of Γ' we pick $(b = 0, b = 1)$ when P_1 deviated and $(b = 1, b = 0)$ when P_0 deviated. Finally in the only subgame corresponding to the history where both players played according to the sampled action at round 1 we pick $(b = 1, b = 1)$ as the equilibrium.

Having eliminated all subgames rooted at round 2 of Γ', we now need to analyse the payoffs of the reduced game tree which is depicted in Table 2. For each P_i, any unilateral deviation from the profile $a = (a_0, a_1)$ results in a lower payoff of $u_i(h) - d_i = u_i(h) - \max_{a \in A}(u_i(a)) + \min_{a \in \text{supp}(\mu)}(u_i(a)) - 1 < u_i(a)$, which is the payoff of playing a_i given P_{1-i} plays a_{1-i} when h is the actual action profile played at round 1 of Γ'.

Thus, the strategy ν is a subgame perfect equilibrium in Γ'.

E.2 Proof of Lemma 2

Lemma 2. *For any $\mu \in \Delta(\bar{A})$, the strategy profile ν is renegotiation-safe in Γ'.*

Proof. First, we consider the possible renegotiations at the first round. Since μ is supported on Pareto optimal action profiles, any renegotiation of strategies in the first round of Γ' results in one player being worse off. Specifically, any

renegotiation improving the utility of player P_i must propose some action profile a' distinct from a and set the opponent's confirmation bit to $b_{1-i} = 1$. Additionally, the confirmation bit of P_i must be also set to $b_i = 1$ as, otherwise, the proposed renegotiation would no be renegotiation safe at the second round. It follows that the existence of profitable renegotiations for player P_i at round one in Γ' is equivalent to the existence of action profile $a' \in \Gamma$ that Pareto dominates a and strictly improves the utility for P_i, but there is no such a' due to the restriction to distributions μ supported solely on Pareto optimal profiles.

In the second round of Γ', the strategy profile $(b_0 = 1, b_1 = 1)$ is renegotiation-safe as any other strategy decreases the payoff for at least one of the players, and so there are no meaningful renegotiations at this round. Thus, since ν is renegotiation-safe at both rounds, it is renegotiation-safe in Γ'.

F Analysis of extortions

F.1 Proof of Lemma 3

Lemma 3. *Naive extortion threats are non-effective without a trusted third party.*

Proof. For the any threat of the form as specified in Γ_{extort} to be effective, the extortioner needs to prove to the victim that if they yield to the threat the extortioner will not burn their deposit, that is, if the victim plays $b_v = 1$ the extortioner will not burn d_v. Suppose the extortioner devised a functionality $\mathcal{F}$ such that the extortioner can use $\mathcal{F}$ to simultaneously release control of the d_v to the victim, and also to ensure the victim pays x to the extortioner. However, this functionality would essentially perform an exchange of actions and thus $\mathcal{F}$ would solve the fair exchange problem which is impossible without a trusted third party. Thus the extortion threat cannot be effective without a trusted third party.

F.2 Proof of Lemma 4

Lemma 4. *Suppose $f_e \leq f_v$. Let $n > 0$ be the first n that satisfies $s_e(n) \geq f_e$. The game terminates at step n with the victim counter extorting the extortioner.*

Proof. The victim plays tit-for-tat: for every x amount the extortioner demands from the victim and y amount locked in the auxiliary smart contract, the victim makes the exact same move, i.e., counter demands x and locks y in their auxiliary smart contract. Since f_e is finite, $\exists n$ such that $s_e(n) \geq f_e$, beyond which the extortioner simply has no more funds to lock into auxiliary smart contracts to make their extortion attacks effective. Since $f_v \geq f_e$, for every auxiliary contract created by the extortioner, the victim can also create a similar auxiliary contract locking in the same amount of funds. When the extortioner runs out of funds, i.e., $s_e(n) \geq f_e$, the extortioner has to respond to the sequence of extortion threats first as the extortioner moves first in the game. Since all the locked funds in the extortioner's auxiliary contracts have to satisfy $y_e^i > x \ \forall i$, the extortioner will have to yield to the victim's counter extortion threat.

F.3 Proof of Proposition 1

Proposition 1. *If we additionally restrict the support of μ to $\mathfrak{A}$, then $\Pi_2^{P.O.\cdot}$ is safe from smart contract extortions.*

Proof. Let $\tilde{a}$ denote the sampled action. Since μ satisfies the conditions underlying Theorem 1 and from Theorem 1, both players will follow the sampled action at round 1 of Γ'. The assumption that both players start off with initial funds d_i which they have to deposit into the smart contract in the setup phase of $\Pi_2^{P.O.\cdot}$, and the solvency constraints of the extortion game implies that at round 1 of Γ' players do not have any reserve funds to conduct an extortion threat with smart contracts. Thus, extortions can only occur in round 2 of Γ' where any player can start an n step extortion game Γ_{extort} as defined in Section 3.1, where the funds of the players at the start of the n step extortion game Γ_{extort} are the payoffs they get from the underlying game Γ when playing $\tilde{a}$. Since both players receive the same payoff, by Lemma 4, any extortion attempt is subject to a tit-for-tat counter extortion threat by the victim and is always successful for the victim. Thus, $\Pi_2^{P.O.\cdot}$ is safe from extortions.

F.4 Proof of Proposition 2

Proposition 2. *If there are action profiles in* $\mathrm{supp}(\mu)$ *that do not give both players the same utility, then $\Pi_2^{P.O.\cdot}$ is unsafe from smart contract extortions.*

Proof. Let $\tilde{a}$ denote the sampled action and let x denote the extortion amount. Following a similar argument to the proof of Proposition 1, extortion threats can only occur at round 2 of Γ'. From Lemmas 5 and 6, in the case where one player, wlog P_1, receives a larger payoff from $\tilde{a}$, i.e., $u_0(\tilde{a}) < u_1(\tilde{a})$, P_1 can start an extortion game that will result in P_0 being extorted an amount of $\frac{u_0(\tilde{a})}{2} + \frac{u_1(\tilde{a})-u_0(\tilde{a})}{2} - \epsilon$ if $F_l > x$ or $\min(F_l, \frac{u_0(\tilde{a})}{2} + \frac{u_1(\tilde{a})-u_0(\tilde{a})}{2} - \epsilon)$ if $F_l < x$. Thus, $\Pi_2^{P.O.\cdot}$ is unsafe from extortions.

G Issues with naive implementation of sampling the random seed

Here we detail how the random sampling of an action profile from $\bar{A}$ by the smart contract in the setup phase of $\Pi_2^{P.O.\cdot}$. The difficulty in sampling a random action profile is ensuring that the seed used for sampling is securely random. In particular, it should not be manipulable by the players participating in $\Pi_2^{P.O.\cdot}$. We first outline a naive but problematic method of obtaining such a seed. A naive way to generate the random seed at time t_0 is for the smart contract to simply hash the current block of the blockchain at t_0 and use the output as a random seed. However, since the sampling time t_0 is fixed in advance by both parties, either party can attempt to manipulate the seed by launching grinding attacks on the blockchain. Specifically, players can try to mine several blocks or

try different subsets of transactions to get a block that would lead to a more favourable seed.

A similar issue could also arise with methods that obtain the random seed via some subset of external users (typically through the mechanism of an external randomness generation game, see for example [16]). Although the external game mechanism by itself might guarantee the secure generation of the seed, composing the external mechanism with our protocol might not necessarily achieve the same security. This is due to the fact that the players in our protocol could potentially communicate and collaborate with the external users, thus changing their incentives.

We thus propose using a secure randomness beacon to sample a random seed. Specifically, our procedure to sample an action a from μ using a random seed from a secure randomness beacon is as follows:

Sampling procedure:
- At time t_0, $\mathcal{SC}$ collects a random seed s from a secure randomness beacon, e.g., NIST beacon.
- $\mathcal{SC}$ samples an action a from μ using the inverse probability integral transform using s.

H Details and analysis of $\Pi_2^{P.O.}$ in the computational setting

H.1 Protocol details

Denote the confirmation message each party P_i, $i \in \{0, 1\}$, wants to send to the smart contract as m_i. Below we detail the time-lock encryption protocol for each P_i as well as the time-lock decryption protocol done by the smart contract (both protocols depicted in Figure 6).

Time-lock encryption protocol:
- P_i generates $N_i = p_i q_i$ where p_i, q_i are randomly selected large primes.
- Each party selects a time parameter τ_i which determines how long the solver would need to solve the puzzle. The time parameter should be selected such that $\tau_i > (t_2 - t_1) \cdot S$ where S is the number of squarings the underlying solver used by $\mathcal{SC}$ can perform (we assume both parties know this information). Thus the total number of squarings would be $T_i = \tau_i \cdot S$.
- P_i picks a random key from the Gen algorithm from any typical symmetric key encryption scheme, i.e., $k_i \leftarrow \mathsf{Gen}(1^n)$. P_i generates the message ciphertext $c_i^m \leftarrow \mathsf{Enc}_{k_i}(m_i)$.
- P_i picks a random x_i from $\mathbb{Z}_{N_i}^*$ and computes $y_i = {x_i}^{2^{T_i}} \bmod N_i$. This can be done efficiently because P_i knows $\phi(N_i) = (p_i - 1)(q_i - 1)$.
- P_i generates key ciphertext $c_i^k = k_i + y_i$.
- P_i sends message tuple $\mathcal{M}_i = (c_i^k, c_i^m, x_i, N_i, T_i)$ to $\mathcal{SC}$ any time between t_1 and t_2.

Time-lock decryption protocol:

- $\mathcal{SC}$ waits until time t_2 to get message tuples from both parties. If any party sends multiple message tuples, $\mathcal{SC}$ keeps the first one and ignores the rest. Any message tuple received after time t_2 is discarded.
- Immediately after t_2, $\mathcal{SC}$ parses $\mathcal{M}_i$ to get the time-lock puzzle element x_i T_i.
- If at any time after t_2 $\mathcal{SC}$ receives a valid key k_i from party i, $\mathcal{SC}$ decrypts c_i to get $m_i = \mathsf{Dec}_{k_i}(m_i)$ and publishes it. Otherwise $\mathcal{SC}$ continues squaring x_i until it gets ${x_i}^{2^{T_i}} \bmod N_i$.

Lemma 7. *The time-lock decryption protocol terminates and takes $O(T_0 + T_1)$ steps.*

Proof. If both parties send their keys to $\mathcal{SC}$ at any point in the decryption protocol the process terminates immediately. If only one party P_i sends their key to $\mathcal{SC}$, SC will decrypt the ciphertext c_i and continue solving the puzzle of P_{1-i}. This will take a maximum of T_{1-i} steps due to the sequential conjecture of time-lock puzzles. If neither party sends their key, $\mathcal{SC}$ will solve both puzzles which will take $T_0 + T_1$ steps.

Remark 1. For the time-lock decryption protocol to terminate by time $t_2+\tau_0+\tau_1$, the solver has to ensure that it continuously solves the puzzles, i.e., it cannot pause or divert its computation.

Remark 2 (On the choice of τ_i in practice). In practice, when replacing simultaneous broadcast of actions using time-lock encryption on the blockchain, players typically choose the times t_0, t_1, t_2 during the setup phase to correspond to specific future blocks on the blockchain. Here, we stress that the lower bound of $((t_2 - t_1) \cdot S)$ on the time parameter τ_i stated in our protocol is a theoretical lower bound in the ideal case where there is not irregularity between block inter-arrival times. In particular, unusually long waiting times between blocks could negatively impact the security of the protocol. Thus, in practice τ_i would typically be chosen to be larger than $(t_2 - t_1) \cdot S$ to account for such potentially long waiting times.

We stress that although the time-lock decryption protocol incurs a computational cost of $O(T_0 + T_1)$ on the smart contract, in practice it could be much smaller if players send their keys to the contract shortly after time t_2. We note that players are actually incentivised to do so as delaying the decryption process would result in their collateral being locked for a longer period, thus incurring a larger opportunity cost.

Let k_i and τ_i denote the security and time parameters respectively of the time-lock puzzle chosen by P_i and n denote the security parameter of the underlying CPA-secure encryption scheme. The next claim shows that under reasonable assumptions, replacing simultaneous broadcast with time lock puzzles preserves the security of our protocol.

Lemma 8. *Assuming the underlying encryption scheme is CPA-secure, the sequential conjecture of time-lock puzzles is true and factorisation is hard, the*

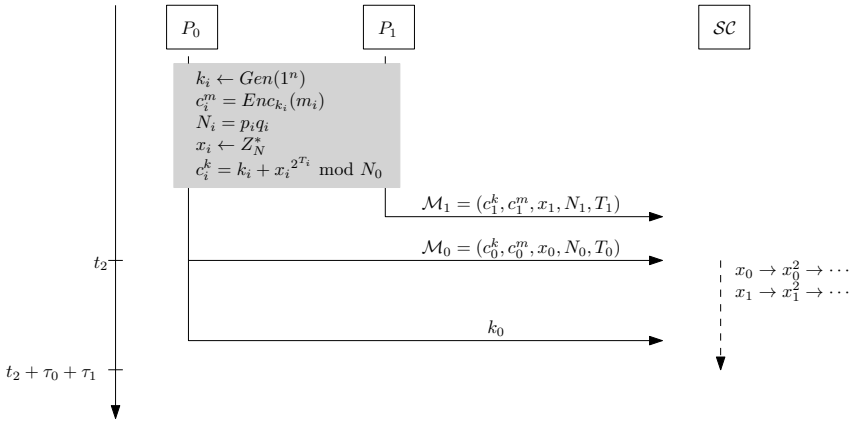

Fig. 6: The time-lock encryption and decryption protocol. Even if P_1 never sends their key k_1 to the smart contract, the smart contract can still compute the key and decrypt c_1 by time $t_2 + \tau_0 + \tau_1$.

probability that a PPT player P_i can retrieve m_{1-i} in significantly less than τ_{1-i} time is negligible in n and k_{1-i}.

Proof. Let p be the probability that a PPT player P_i can retrieve m_{1-i} in significantly less than τ_{1-i} time. This can only happen in 2 ways: either P_i breaks the underlying encryption scheme, or P_i solves P_{1-i}'s puzzle in significantly less than T_{1-i} steps. The former can only happen with probability $\mathsf{negl}(n)$ where n is the security parameter of the scheme as the encryption scheme is CPA-secure. The latter is as hard as factorising N_{1-i} and thus can only happen with probability $\mathsf{negl}(k_{1-i})$. Using the union bound, we get $p < \mathsf{negl}(n) + \mathsf{negl}(k_{1-i})$ which is also negligible.

H.2 Proof of Theorem 2

Theorem 2. *If we impose the same restrictions on μ as in Theorem 1, the strategy $\nu = (a, (b = 1, b = 1))$ is a computational renegotiation-safe subgame perfect equilibrium in the two-player, two-round extensive form game $\tilde{\Gamma}'$ induced by $\tilde{\Pi}_2^{P.O.}$.*

We first note that we can extend the definition of renegotiation-safe strategies to the computational setting by adding a negligible term to the lower bound the renegotiated strategy profile σ' has to satisfy in Definition 2, i.e., $u_i([\sigma, r, \sigma']_i) > u_i(\sigma) - (\mathsf{negl}(n) + \mathsf{negl}(k_0) + \mathsf{negl}(k_1))$.

Proof. Let p be the probability that a PPT player P_i either breaks the underlying encryption scheme or solves P_{1-i}'s puzzle in significantly less than T_{1-i} steps. From Lemma 8, p is negligible. We first note that the game $\tilde{\Gamma}'$ induced by $\tilde{\Pi}_2^{P.O.}$ corresponds exactly to the game Γ' induced by $\Pi_2^{P.O.}$ with just the

following differences in utilities at the leaves of the game tree: with an overwhelming probability of $1-(2p+p^2)$, the utilities at the leaves of $\tilde{\Gamma}'$ are the same as in Γ'. This corresponds to the case where no player breaks the underlying cryptographic primitives. In the case where at least one player breaks the primitives, any change to their utility can only come in the second round of $\tilde{\Gamma}'$ during the redistribution of payoffs or from an extortion. This is upper bounded by the maximum utility any player can get in the underlying game Γ and we denote this by $u_{max} = \max(\max_{a \leftarrow \mu} u_0(a), \max_{a \leftarrow \mu} u_1(a))$.

Since u_{max} is a constant, any change to the expected utilities in the second round of $\tilde{\Gamma}'$ is at most $l \cdot (2p+p^2) \cdot u_{max}$ where l is the number of leaves of the game tree corresponding to $\tilde{\Gamma}'$. This is negligible and hence the Nash equilibria chosen in the proof of Lemma 1 during the second round of Γ' are computational Nash equilibria in $\tilde{\Gamma}'$. Similarly, since the reduced game tree of Γ' uses the payoffs from the Nash equilibria at the subgames at round 2 of Γ', the Nash equilibrium in the reduced game tree of Γ' is also a computational Nash equilibrium in $\tilde{\Gamma}'$. Thus, $\nu = (a, b = 1)$ is a computational subgame perfect equilibrium in $\tilde{\Gamma}'$.

To show computational renegotiation-safeness, we again note that any difference in utilities from Γ' only happen in the second round of $\tilde{\Gamma}'$, so the analysis in Lemma 2 for the first round follows for $\tilde{\Gamma}'$. Similar to the above analysis, since any change in the expected utilities in the second round of $\tilde{\Gamma}'$ is at most $l \cdot (2p+p^2) \cdot u_{max}$, hence negligible, we can conclude that the strategy $\nu = (a, b = 1)$ is computationally renegotiation-safe in the second round of $\tilde{\Gamma}'$ and hence computationally renegotiation-safe in $\tilde{\Gamma}'$.