An Incentive-Compatible Smart Contract for Decentralized Commerce

Nikolaj I. Schwartzbach
Dept. of Computer Science
Aarhus University
Aarhus, Denmark

***Index Terms*—game theory, smart contract, electronic commerce**

***Abstract*—We propose a smart contract that allows two mutually distrusting parties to transact any non-digital good or service on a blockchain. The contract acts as an escrow and settles disputes by letting parties wager that they can convince an arbiter they were the honest party. We analyze the contract as an extensive-form game and prove that the contract is secure in a strong game-theoretic sense if and only if the arbiter is biased in favor of honest parties. We show this is inherent to any contract that achieves game-theoretic security for interesting trades. We consider a generalization of the contract with different ways of paying back the wagers, and we can instantiate it to make a tradeoff between security and the size of the wager. By relaxing the security notion such that parties have only weak incentive to behave honestly, we can replace the arbiter by a random coin toss protocol. We implement the contract in Ethereum and estimate the amortized cost of running the contract as 2-3 USD for the seller and 4-5 USD for the buyer.**

I. Introduction

A fundamental problem of electronic commerce is ensuring both ends of the trade are upheld: an honest seller should always receive payment, and an honest buyer should only pay if the seller was honest. Traditionally, this is ensured by introducing a trusted intermediary who holds the payment in escrow until the trade has completed, after which it releases the funds to the seller. It typically requires the parties not to be anonymous, to enable either party to hold the other party accountable in case of fraudulent behavior, and potentially subject to legal repercussions. This, in conjunction with reputation systems, has proved to be an effective means to honest and efficient trading, as evidenced by the enormous market cap of online marketplaces such as Amazon or Alibaba. However, this relies on being able to trust the intermediary to behave honestly: while the intermediary has a strong incentive to maintain a good reputation, this does not address the fundamental issue from a cryptographic point of view. Besides obvious privacy concerns, a central marketplace also has an incentive to engage in monopolistic behavior, such as removal of competitors' products or differential pricing based on customer demographics to the extent that it remains undetected [1].

Recent years have seen the creation of darknet markets that take advantage of cryptocurrency and mix networks to provide decentralized and somewhat anonymous trade of goods and services. They arguably solve some issues with central marketplaces, but in doing so, also enable black market/criminal activity to remain relatively unchecked. The most infamous darknet market was "Silk Road", known for selling illicit goods such as drugs, weapons, and fake passports. It operated from February 2011 until the authorities seized it in October 2013, and the developer, Ross Ulbricht, sentenced to double life imprisonment. But this is a rarity: because of the anonymous nature of the markets, it is often difficult to prosecute individuals, and many convictions of buyers are based on circumstantial metadata such as credit card transactions to purchase cryptocurrency of similar size. However, most darknet markets remain inherently centralized, in that all data and escrowed funds are processed directly by the market itself, essentially at its mercy. It requires buyers and sellers to trust both the benevolence and competence of the market, a trust which is at best misplaced and at worst disastrous in consequence. There are many examples of prominent darknet markets being hacked, and all funds held in escrow stolen, or of the operators of the market themselves performing an *exit scam*, i.e. suddenly stealing the funds in escrow and subsequently closing the market. It is often difficult, if not impossible, to recover the stolen funds and hold anyone accountable [2].

In this paper, we consider a seller who wants to sell an item to a buyer without having to rely on a trusted third party. We assume both parties are rational and have shared access to a blockchain that allows them to deploy smart contracts that can exchange cryptocurrency. Our goal is to replace the trusted third party with a smart contract, such that parties can be trusted to complete their end of the trade. As the parties are rational, we want to prove they maximize their utility by behaving as they should. We ask, *can we design a smart contract to facilitate decentralized trading of non-digital goods and services in a way that provably ensures honest behavior in rational agents?*

A. Our results

We propose a smart contract for escrow of funds that enables any two parties to engage in the trade of a physical good or service for cryptocurrency. The contract relies on an *arbiter* that is invoked only in the case of a dispute. The purpose of the arbiter is to distinguish the honest party from the dishonest party. Either party may issue a dispute by making a "wager" of size λ that they will win the arbitration: the winner is repaid their deposit and the funds held in escrow. We prove that both buyer and seller are incentivized to behave honestly if and only if the arbiter is biased in favor of honest parties. Specifically, let γ be the "error rate" of the arbiter: then we show there is a value of λ such that the contract has strong game-theoretic security if and only if $\gamma < \frac{1}{2}$. This is not a particularity of our contract: we show this is inherent to any contract that achieves game-theoretic security for interesting trades. By instead considering a weaker notion of security where parties do not have strict incentive to behave honestly, we can use a random coin flip protocol as the arbiter. However, the contract remains secure in a strong way against risk-averse adversaries. We sketch a simple construction based on Blum's coin toss protocol.

The contract can be run on any blockchain that supports smart contracts (such as Ethereum). As a result, many properties (anonymity, efficiency, etc.) of the contract are inherited from the corresponding blockchain. We feature a discussion of different ways to instantiate the smart contract. In particular, the contract can be used in a manner that complies with current laws and regulations by using a blockchain with revocable anonymity: a party who takes part in distributing illicit goods can be deanonymized by the courts, while all other parties remain anonymous. This would allow for a kind of certification or blue-print of marketplaces based on smart contracts even if they are essentially anonymous, so long as the underlying blockchain uses revocable anonymity. We implement the contract in Solidity and evaluate it on the Ethereum blockchain. We find that the amortized cost of running the contract is 2-3 USD for the seller and 4-5 USD for the buyer.

B. *Related work*

A variety of solutions have been proposed for replacing the trusted third party by a smart contract in so-called atomic swaps. Most academic work has focused on digital goods, the delivery of which can be deterministically determined.

Dziembowski, Eckey and Faust propose a protocol, called *FairSwap* [3], with essentially optimal security: the goods are delivered to the buyer if and only if the seller receives the money. Their solution relies on cryptography and assumes the goods can be represented as a finite field element. As a result, their protocol does not apply in any meaningful way to physical goods. It seems unlikely we can achieve this notion of security for non-digital goods due to a fundamental difference between the physical and the digital world.

Asgaonkar and Krishnamachari propose a smart contract for the trade of digital goods [4]: both parties deposit funds *a priori* (a *dual-deposit*) which is only refunded if the trade was successful. They prove that the honest strategy is the unique subgame perfect equilibrium for sufficiently large deposits. Like FairSwap, their solution only works for digital goods as it requires a hash function to verify delivery of the item.

Witkowski, Seuken and Parkes consider the setting of escrow in online auctions [5]. Their idea is to pay some of the buyers a rebate to offset their expected loss from engaging in a transaction with the seller. Whether a buyer is paid a rebate depends on the reports of other buyers. They prove that the seller has a strict incentive to be honest, while the buyers are only weakly incentivized to do so. They show that strict incentives for the buyers are possible if the escrow has distributional knowledge about the variations in seller abilities, based on a *peer prediction* method. Unfortunately, their solutions rely on a somewhat idealized setting in which there are many buyers concurrently transacting with the same seller, as otherwise buyers and/or sellers may have an incentive to collude, thus breaking security. Besides, it is not obvious how to apply their work to a non-auction setting.

Outside academic circles, there are several proposed solutions, of which the most promising are Kleros [6], [7] and OpenBazaar [8]. They are both blockchain-based and as such provide some level of decentralization. Unfortunately, the dispute resolution of OpenBazaar remains centralized in a sense, since all moderation is done by an agreed upon moderator, requiring both buyer and seller to trust the moderator. From a cryptographic point-of-view this only serves to move the problem of having to trust the seller to having to trust the moderator. The dispute resolution of Kleros is more sophisticated, in that arbitration is done by a decentralized court where jurors can opt-in on a case-by-case basis. Jurors who vote in accordance with the majory decision are rewarded with money, while jurors who vote differently are penalized. They argue security by the use of focal points, defined as the strategy people choose in the absence of communication: jurors will act honestly because they expect other jurors to do so. Unfortunately, no empirical study of Kleros has been published, so whether the focal point of Kleros is "truth" remains conjecture at this point. Besides, neither system has any formal analysis of correctness or security, and thus fall short in rigorously solving the buyer and seller's dilemma. To the best knowledge of the author, there is no "truly decentralized" market with game-theoretic security at the time of writing.

II. The basic contract

In this section, we describe our contract for the trade of non-digital goods and services. We consider a buyer B who wants to purchase an item it from a seller S. The item can be a physical good or a service. The item is sold for a price of x, and has a "perceived value" to the buyer of $y > x$, while the seller perceives the value at $x' < x$. From a game-theoretic point of view, we have to assume $y > x > x'$, as otherwise neither buyer nor seller has incentive to engage in the transaction. The item it is *non-digital* which means it has to be shipped through a physical channel "off-chain". See Fig. 1 for an illustration. By definition, no computer program can rigorously determine whether or not it was physically delivered to the buyer. This is a fundamental difference between the digital and the physical world. We assume both parties have access to a blockchain, which for our purposes is a shared data structure that allows both parties to deploy a smart contract π that can maintain state, respond to queries, and transfer funds. Unlike a human third party, the smart contract can be guaranteed

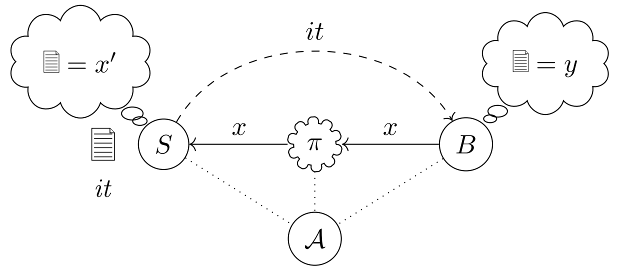

Fig. 1. A seller S and a buyer B engaged in the transaction of the non-digital good it, using a smart contract π and arbiter $\mathcal{A}$. The item is sold for x funds and has a perceived value of $y > x$ to B. The money x is transferred from B to S through the contract π. The dashed line is a unidirectional "off-chain" channel, through which it can be sent. The dotted lines indicate that $\mathcal{A}$ is only invoked in case of disputes.

to behave honestly due to the security of the underlying blockchain. For simplicity, we assume the blockchain is secure and incorruptible, and consider only attacks on the contract itself. For now, we assume transaction fees are negligible compared to the items being transacted, such that they can be disregarded entirely. We will dispense with this assumption later.

The contract is parameterized by an *arbiter* $\mathcal{A}$ which is a protocol invoked in case of disputes: its purpose is to distinguish the honest party from the dishonest party. We denote by γ the error rate of the arbiter. In the case of digital goods, cryptography allows us to get $\gamma = 2^{-\kappa}$ for any κ which has been exploited in previous work [3], [4]. We assume each party holds an estimate of $\gamma > 0$ that they provide as input to the contract. This value might be established empirically, though it likely depends on the nature of the goods transacted. A naive solution is to invoke the arbiter at every transaction to determine whether or not the seller should be paid. However, this is impractical because invoking the arbiter is potentially expensive; we desire a solution that only invokes the arbiter when necessary. We parameterize the contract by a *wager constant* $\lambda > 0$. The contract proceeds as follows: both parties sign a contract committing to making the trade, and B places x money in escrow. S then delivers it to B "off-chain" who then notifies the smart contract to transfer the funds in escrow to S, thus terminating the contract. If S does not deliver it to B, then B can trigger a dispute by placing a "wager" of size λ that they can convince the arbiter that they were the honest party. If S does not respond (or forfeits), it is assumed it was not delivered to B and the contract refunds $x + \lambda$ funds to B. However, a dishonest buyer may trigger the dispute phase even when they received it. In this case, the honest S may counter the wager by also placing a wager of size λ that they will win the arbitration. Of course, a dishonest S may also counter the wager. If both parties counter, the arbiter is invoked and chooses a winner among them. The winner is repaid $x + \lambda$, while the loser receives nothing. We can use the leftover λ to compensate the arbiter for their time. We handle crashing by having timeouts in the contract in a way that favors the party that did not crash; a buyer that crashes is assumed to have received it. Likewise, a seller who fails to respond to a dispute is assumed to forfeit.

III. Preliminaries

We give a brief recap of some basic game-theoretic concepts for the purpose of self-containment. For details, see [9]–[11].

Definition 1. *A* finite game G in extensive form with perfect information *consists of:*

- *A set of n players $\mathsf{P}_1, \mathsf{P}_2, \dots \mathsf{P}_n$.*
- *A rooted tree T. We denote by L the set of leafs in T, i.e. nodes with outdegree 0.*
- *A mapping $u : L \to \mathbb{R}^n$, that for each leaf ℓ gives the utility $u_i(\ell)$ of player P_i.*
- *A partition of the non-leafs into n sets, one for each player. We say a node is owned by a player if it belongs to their partition.*

A game is played by starting at the root and recursively letting the player who owns the node choose a child to descend into. At some point we reach a leaf ℓ, after which player P_i is given $u_i(\ell)$ utility. A subgame *of G is a subtree that is also a finite game in extensive form.* ◇

Definition 2. *A* strategy *s_i for a player P_i in a game G is a distribution over the children of each node owned by P_i in G. If the strategy chooses a child with probability 1 in each node owned by P_i, we say the strategy is* pure. *A* strategy profile *$s = (s_1, s_2, \dots s_n)$ is a strategy for each player, and defines a distribution on the output leafs. If C is a set of players, we may write $s = (s_C, s_{-C})$. We denote by $u_i(s)$ the expected utility of player P_i when playing strategy profile s.* ◇

Definition 3. *A strategy profile s^* is a t-*resilient (Nash) equilibrium *for G if for every coalition $C \subseteq \{1, 2, \dots n\}$ such that $|C| = t$, and every strategy s_C, and every $i \in C$,*

$$u_i(s^*_C, s^*_{-C}) \geq u_i(s_C, s^*_{-C})$$

*If in addition, s^* is a t-resilient equilibrium for every subgame of G, we say s^* is a t-*resilient subgame perfect equilibrium *(SPE). It the above formula only holds with a gap such that $u_i(s^*_C, s^*_{-C}) + \varepsilon \geq u_i(s_C, s^*_{-C})$ for some $\varepsilon > 0$, we say the equilibrium is an ε-*equilibrium. ◇

Our security definition will be a refinement of what is known as an evolutionary stable strategy (ESS). They are studied in biology as a refinement of the Nash equilibrium for determining which strategies are stable wrt. natural selection. An ESS is an equilibrium such that no other equilibrium can replace it as the dominant strategy in a population. We use a slight generalization of the definition by Thomas [11]:

Definition 4. *A strategy profile s^* is a (*t-resilient) evolutionary stable strategy *(ESS) if it holds that:*

- *(*Completeness*). s^* is a t-resilient equilibrium.*
- *(*Soundness*). For every coalition C of size t, and every strategy $s \neq s^*$, and every $i \in C$, $u_i(s^*_C, s_{-C}) > u_i(s_C, s_{-C})$.* ◇

IV. Game-theoretic security

We now turn to instantiate the basic smart contract to achieve security in a game-theoretic sense. Unlike the standard cryptographic model, where parties can be partitioned into honest and dishonest parties, instead, we assume all parties are *rational*, meaning they seek to maximize their own utility with no concern for the intended behavior of the protocol designer: a rational party will at each point in the execution of the protocol choose the action that maximizes their expected utility. We say the protocol is secure when the maximal utility is achieved when a party behaves honestly.

We consider an n-party protocol π where each party P_i has a set $\mathcal{S}_i$ of possible strategies, of which there is a unique *honest strategy* $s_i^* \in \mathcal{S}_i$. We define $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_n$ as the set of strategy profiles, and let $s^* = (s_1^*, \ldots, s_n^*) \in \mathcal{S}$ be the unique honest strategy profile. For our purposes, subgame perfection is likely not sufficient in itself: if the incentive to choose s^* is too small, then there may be other reasons to deviate not captured by the utilities of the game, say for sport or for revenge. If the difference is sufficiently large (say $> \varepsilon$) then we say the protocol is secure in a game-theoretic sense against ε-deviating rational adversaries.

Definition 5 (Strong security). *Let π be an n-party protocol with strategy space $\mathcal{S}$, where $s^* \in \mathcal{S}$ is the unique honest strategy profile. We say π has* t-resilient, ε-strong game-theoretic security *if the following is satisfied:*

- (Completeness) - *s^* is the unique t-resilient subgame perfect equilibrium.*
- (Soundness) - *For every $s \neq s^*$, and every coalition C of size t, and every $i \in C$: $u_i(s_C^*, s_{-C}) - \varepsilon \geq u_i(s_C, s_{-C})$.*

The following is an immediate consequence:

Lemma 1. *If π has t-resilient ε-strong game-theoretic security for some $\varepsilon > 0$ then s^* is a t-resilient evolutionary stable strategy.* ☐

A feature of this definition is its resilience to collusion in separate instances of the game: the games are essentially independent so we may suppose they are played in some order. We now ask if the resulting finite multi-stage game is still secure. Fortunately, this turns out to be the case due to our security definition since subgame perfection satisfies what is known as the *one-shot deviation principle* (in finite games): security in a single game implies security security in the repeated setting. To see why, we proceed using backwards induction. In the last game, the rational buyer will act honestly by security of the protocol; there is no avenue to deviate from the honest strategy. Knowing this, the second-to-last buyer must also act honestly, and so on, showing that rational parties will not collude.

A. Analysis of contract

To analyze the contract from a game-theoretic perspective, we consider the contract as an extensive-form game and draw the corresponding game tree (seen in Fig. 2). The payoff for

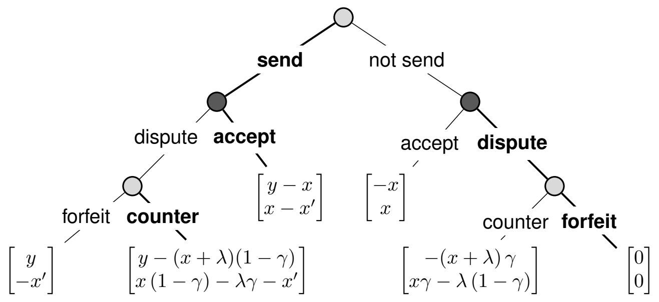
send
not send
dispute
accept
accept
dispute
forfeit
counter
counter
forfeit
$\begin{bmatrix} y - x \\ x - x' \end{bmatrix}$
$\begin{bmatrix} -x \\ x \end{bmatrix}$
$\begin{bmatrix} y \\ -x' \end{bmatrix}$
$\begin{bmatrix} y-(x+\lambda)(1-\gamma) \\ x\,(1-\gamma)-\lambda\gamma-x' \end{bmatrix}$
$\begin{bmatrix} -(x+\lambda)\,\gamma \\ x\gamma-\lambda\,(1-\gamma) \end{bmatrix}$
$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$

Fig. 2. Game tree of the smart contract after both parties have accepted the transaction. The first coordinate is the buyer payoff and the second is seller payoff. Light nodes are seller actions; dark nodes are buyer actions. The heavy edges denote honest actions.

each party is defined as their expected change in funds, where for simplicity we have explicitly omitted transaction fees. As an example, consider a dispute between a dishonest buyer and an honest seller. The buyer has earned y value since the seller was honest. The buyer may lose the arbitration with probability $1-\gamma$, in which case they lose $x+\lambda$, for an expected payoff of $y-(x+\lambda)(1-\gamma)$. Likewise, the seller receives their payment of x with probability $1-\gamma$ and loses λ with probability γ, for an expected payoff of $x\,(1-\gamma)-\lambda\gamma-x'$. The other cases are similar and are summarized in Fig. 2.

Lemma 2. *There is a value of λ such that the contract is complete if and only if the arbiter is biased in favor of honest parties.*

Proof. We proceed using backwards induction in the game tree. We see that the honest actions yield a strictly larger payoff if and only if the following inequalities are satisfied:

$$x\,(1-\gamma) - \lambda\gamma - x' > -x' \tag{1}$$

$$0 > x\gamma - \lambda\,(1-\gamma) \tag{2}$$

$$y - x > y - (x+\lambda)(1-\gamma) \tag{3}$$

Eq. (1) says that an honest seller counters a dispute from a dishonest buyer. Eq. (2) says that a dishonest seller forfeits a dispute from an honest buyer. Eq. (3) says that a buyer will not dispute when they received it. In addition, we need $0 > -x$ and $y-x>0$ but these come from the problem statement. From Eq. (1) we get $\lambda < x\,(\frac{1-\gamma}{\gamma})$, while Eq. (2) yields $\lambda > x\,(\frac{\gamma}{1-\gamma})$. From Eq. (3) we also get $\lambda > x\,(\frac{\gamma}{1-\gamma})$ but this is equivalent to Eq. (2). In summary, any value of λ that achieves completeness must satisfy:

$$x\left(\frac{\gamma}{1-\gamma}\right) < \lambda < x\left(\frac{1-\gamma}{\gamma}\right)$$

But this can only be true when $\gamma < \frac{1}{2}$. ☐

Lemma 3. *Let $\varepsilon > 0$ and suppose $y - \varepsilon \geq x \geq \varepsilon$. Then there is a value of λ such that the contract is ε-sound if and only if $\gamma < \frac{1}{2}$ and $\varepsilon \leq x\,(1-2\gamma)$.*

Proof. In any dishonest strategy profile, one of the parties must choose a dishonest action. If we can show all honest actions have $\geq \varepsilon$ more utility than the dishonest actions, then the contract is ε-sound. This gives the same equations as Lemma 2, except we subtract ε from the left-hand side and swap $>$ for $\geq$. Solving for λ, the first equation gives $\lambda \leq \frac{x(1-\gamma)-\varepsilon}{\gamma}$. The second gives $\lambda \geq \frac{x\gamma+\varepsilon}{1-\gamma}$, while again the third is equivalent to the second. We also need $-\varepsilon \geq -x$ and $y - x \geq \varepsilon$ but these are given in the problem statement. In summary, any values of λ, ε must satisfy:

$$\frac{x\gamma+\varepsilon}{1-\gamma} \leq \lambda \leq \frac{x(1-\gamma)-\varepsilon}{\gamma}$$

Again, we must have $\gamma < \frac{1}{2}$ since $\varepsilon > 0$, while the latter condition can be established by solving for ε. □

Theorem 1. *The contract has $x(1-2\gamma)$-strong game-theoretic security whenever $\gamma < \frac{1}{2}$ and $\lambda = x$.*

Proof. Since $\gamma < \frac{1}{2}$ the conditions for completeness are satisfied for $\lambda = x$. For soundness, let $\varepsilon = x(1-2\gamma)$. We choose a value of λ that satisfies the lower bound: $\lambda \geq \frac{x\gamma+\varepsilon}{1-\gamma} = \frac{x\gamma+x(1-2\gamma)}{1-\gamma} = x$. In particular, for $\lambda = x$ the equation is always true. □

V. The generalized contract

In this section, we consider a generalization of the previous contract that allows us to obtain various tradeoffs between security and wager size. We show that the lower bound of $\gamma < \frac{1}{2}$ is inherent to any contract that achieves game-theoretic security for 'interesting trades'. We define an interesting trade as a trade where parties have a net increase in utility if they are successful in cheating, compared to being honest. If a trade is not interesting, security is trivial.

Theorem 2. *Any protocol that achieves completeness for interesting trades can only be complete if it invokes an arbiter who is biased in favor of honest parties.*

Proof. In any interesting trade, a rational party will always choose to be dishonest unless there is some chance that the strategy gives smaller utility. This necessitates the use of some mechanism $\mathcal{A}$ that determines if a party was dishonest. A rational dishonest party will never reveal themselves if they lose utility by doing so, so $\mathcal{A}$ needs to be external to the parties in the protocol. We call such a mechanism an *arbiter*. We can assume the arbiter is only invoked if one of the parties were dishonest, and we will assume $\mathcal{A}$ outputs a single bit determining whether a fixed party (say the seller) were the dishonest party. If a party is deemed dishonest by the arbiter, we say they are the 'winner'; otherwise they are the 'loser'. We let γ be the error rate of $\mathcal{A}$, i.e. the probability that the loser were honest. Let ω, ℓ be functions such that the winner is paid ω money and the loser is paid ℓ money. We have to assume that $\omega > \ell$ such that winning is preferred over losing. Now, consider a seller who has to decide whether or not to counter a dispute from the buyer. Regardless of whether the seller is honest or not, they have to decide whether to forfeit or counter. If the seller is honest we want them to counter the dispute, i.e. $\omega(1-\gamma) + \ell\gamma > 0$. If the seller is dishonest we want them to forfeit, i.e. $\omega\gamma + \ell(1-\gamma) < 0$. That is,

$$\omega\gamma + \ell(1-\gamma) < \omega(1-\gamma) + \ell\gamma.$$

Since we have $\omega > \ell$ this can only be true for $\gamma < \frac{1}{2}$. □

A. Affine rebate functions

In the following we let α be a constant such that the winner is paid back $\alpha\lambda$ money. Naturally, we must have that $\alpha \geq 0$ as the winner cannot lose more than they have wagered. Also, we must have $\alpha \leq 2$ to prevent the contract from minting money. Note that the original contract is a special case where $\alpha = 1$.

Lemma 4. *The contract is complete if and only if $\gamma < \frac{1}{2}$, and $\lambda > x\left(\frac{\gamma}{1-\alpha\gamma}\right)$, and either 1) $\alpha \geq \frac{1}{1-\gamma}$; or 2) $\lambda < x\left(\frac{1-\gamma}{1-\alpha(\gamma-1)}\right)$.*

Proof. In order to show completeness, we again proceed using backwards induction. As before, we only need to consider a seller faced with a dispute, as this implies the other cases. That is, we have the following two equations:

$$(x+\alpha\lambda)(1-\gamma) - \lambda > 0 \tag{4}$$

$$0 > (x+\alpha\lambda)\gamma - \lambda \tag{5}$$

For any choice of α, in Eq. (5) by isolating λ we get the lower bound of $\lambda > x\left(\frac{\gamma}{1-\alpha\gamma}\right)$ from the statement. For the other clause, we can write Eq. (4) differently:

$$x(1-\gamma) > \lambda(1-\alpha(1-\gamma)) \tag{6}$$

In order to isolate λ we need to divide with $\alpha(1-\gamma)$. When $\alpha < \frac{1}{1-\gamma}$ this number is positive which means we retain the direction of the inequality, thus giving the upper bound of $\lambda < x\left(\frac{1-\gamma}{1-\alpha(\gamma-1)}\right)$. However, when $\alpha \geq \frac{1}{1-\gamma}$, we divide by a negative number, thus flipping the inequality and giving a trivial lower bound of $\lambda > -x\left(\frac{1-\gamma}{\alpha(1-\gamma)-1}\right)$, showing the desired statement. □

Theorem 3. *The contract has (maximal) ε-strong game-theoretic security if and only if $\gamma < \frac{1}{2}$ and one of the following conditions are established:*

1) $\alpha = 2$*; and* $\lambda \geq \frac{x\gamma+\varepsilon}{1-2\gamma}$.
2) $\frac{1}{1-\gamma} < \alpha < 2$*; and* $\varepsilon \geq x\left(\frac{1-2\gamma}{2-\alpha}\right)$*; and* $\lambda \geq \frac{x\gamma+\varepsilon}{1-\alpha\gamma}$.
3) $\alpha = \frac{1}{1-\gamma}$*; and* $\varepsilon = x(1-\gamma)$*; and* $\lambda = x\left(\frac{1-\gamma}{1-2\gamma}\right)$.
4) $\alpha < \frac{1}{1-\gamma}$*; and* $\varepsilon = x\left(\frac{1-2\gamma}{2-\alpha}\right)$*; and* $\lambda = x\left(\frac{1}{2-\alpha}\right)$.

Proof. Again, it suffices to consider a seller faced with a dispute. This leads to the following two inequalities:

$$(x+\alpha\lambda)(1-\gamma) - \lambda - \varepsilon \geq 0 \tag{7}$$

$$-\varepsilon \geq (x+\alpha\lambda)\gamma - \lambda \tag{8}$$

Note that since we assume $\varepsilon > 0$, this automatically implies completeness. We now consider different values of α, choose the maximum permitted value of ε to maximize security, and solve for λ. Note that we are intentionally leaving out some cases where ε is small.

1) When $\alpha = 2$, Eq. (8) gives a lower bound of $\lambda \geq \frac{x\gamma+\varepsilon}{1-2\gamma}$, while as in the proof for completeness Eq. (7) gives a trivial lower bound.
2) When $\frac{1}{1-\gamma} < \alpha < 2$, equations Eq. (7) and Eq. (8) give lower bounds of:
$$\lambda \geq \frac{x\gamma + \varepsilon}{1 - \alpha\gamma} \qquad \lambda \geq \frac{x\,(1-\gamma) - \varepsilon}{1 - \alpha + \alpha\gamma}$$
When $\varepsilon \geq x\left(\frac{1-2\gamma}{2-\alpha}\right)$ the lower bound from Eq. (7) is strongest. Since we choose the *maximal* value of ε, i.e. not $\varepsilon < x\frac{1-2\gamma}{2-\alpha}$, this gives the desired bound.
3) When $\alpha = \frac{1}{1-\gamma}$, Eq. (7) gives an upper bound on the security parameter $\varepsilon \leq x(1-\gamma)$. Similarly, Eq. (8) gives a lower bound of $\lambda \geq \frac{(1-\gamma)(x\gamma+\varepsilon)}{1-2\gamma}$. Choosing the maximum $\varepsilon = x\,(1-\gamma)$ and substituting gives the desired result.
4) When $\alpha < \frac{1}{1-\gamma}$, Eq. (7) gives an upper bound of $\lambda \leq \frac{x\,(1-\gamma)-\varepsilon}{1-\alpha+\alpha\gamma}$, while Eq. (8) gives a lower bound of $\lambda \geq \frac{x\gamma+\varepsilon}{1-\alpha\gamma}$. This means there is a value of λ such that ε-soundness is satisfied if and only if:
$$\frac{x\gamma + \varepsilon}{1 - \alpha\gamma} \leq \frac{x\,(1-\gamma) - \varepsilon}{1 - \alpha + \alpha\gamma}$$
The maximal value of ε satisfying this equation is $\varepsilon = x\left(\frac{1-2\gamma}{2-\alpha}\right)$ which solves to $\lambda = \frac{x}{2-\alpha}$, showing the desired. □

B. Tradeoffs

We now have a characterization of different ways of instantiating the contract, allowing us to reason about pros and cons of different choice of parameters. We consider a few special cases: Suppose that, in addition to receiving their own wager back, the winner also receives the loser's wager, i.e. the generalized contract with $\alpha = 2$. By Theorem 3, we have no upper bound on ε, allowing us to get arbitrarily high security by making the wager sufficiently large. The downside to this is that it results in larger wagers: suppose we let $\varepsilon = x\,(1-2\gamma)$, the maximum value in the old contract, then the new wager is:
$$\lambda = \frac{x\gamma + x\,(1-2\gamma)}{1-2\gamma} = x + \frac{x\gamma}{1-2\gamma} > x$$
which is always larger than the old wager. This is natural in a sense: since we expect to win back the wager by disputing, the wager needs to be larger to offset the increased incentive to issue a false dispute.

Corollary 1. *With a winner rebate of size λ, the contract has ε-strong security (for any $\varepsilon > 0$) whenever $\gamma < \frac{1}{2}$ and $\lambda = \frac{x\gamma+\varepsilon}{1-2\gamma}$.* □

Instead, consider what happens the wager is withheld even for the winning party, i.e. letting $\alpha = 0$. This naturally requires $\lambda < x$ since otherwise there would be no incentive to dispute. We again refer to Theorem 3 which gives the following result:

Corollary 2. *When the contract withholds all wagers, it has $\frac{1}{2}x\,(1-2\gamma)$-strong game-theoretic security when $\gamma < \frac{1}{2}$, and $\lambda = \frac{1}{2}x$.* □

It is not hard to see (referring to Theorem 3) that this is the minimal value of λ we can use if we want to maximize the security of the protocol. It seems our construction requires $\lambda = \Omega(x)$. This is natural in a sense: if λ were a constant, by increasing x, at some point the expected utility from attempting to cheat would outweigh the cost of losing the wager.

Invoking the arbiter is not free. This is the very motivation behind our contract: if the arbiter were free and better than random, we could trivially invoke it in every purchase to determine who should receive the money. However, this unfairly punishes honest parties by requiring them to pay for an expensive and unnecessary arbitration. Still, we have to compensate the arbiter somehow. We can accomplish this by varying α such that the left over funds equal the price of the arbitration. If P is the price of the arbitration, we can accomplish this by letting $\alpha = 2 - P/\lambda$. This works whenever $P > 2\lambda$ such that the total wager exceeds the price of paying for the arbitration. We may instantiate this in various ways by referring to Theorem 3.

VI. Practical considerations

In this section we consider various issues that arise when implementing the contract in practice.

A. Transaction fees

Our analysis assumes transaction fees are negligible, which is not the case in practice. In this section, we consider adding transaction fees to our model. Doing so in general is tricky business and is very specific to the implementation and the blockchain of choice. Instead, we adopt a simplified approach where playing a move in the game tree has a unit cost of τ for some $\tau > 0$, the only exception being the default action in case of timeouts: a player can always time out to choose the default action at zero cost. For simplicity, we let $\alpha = 1$.

Lemma 5. *With transaction fees of size τ, the contract is complete if and only if the arbiter is biased in favor of honest parties, the transaction fee is bounded $\tau < x\,(1-\gamma) - \lambda\gamma$, and the item is of sufficient value, $x - x' > \tau$.*

Proof. We proceed using backwards induction in the game tree. It is not hard to see we still need $\gamma < \frac{1}{2}$. Consider a seller faced with a dispute. When they are dishonest their incentive to be honest is increased by τ, while the converse is true when they are honest. This yields $\tau < x\,(1-\gamma) - \lambda\gamma$. Now consider a buyer. If they did receive the item, their incentive to accept has only increased by τ. If they did not receive the item, their added cost of τ for issuing a dispute must outweigh the size of the payment. This means we must have $x > \tau$. Finally, consider a seller deciding whether to send or not. If they do not send they incur a cost of 0, while accepting gives $x - x' - \tau > 0$. □

Lemma 6. *With transaction fees of size τ, the contract is ε-sound only when $\varepsilon \leq x(1-2\gamma)-\tau$, and the transaction fee is bounded $\tau < x(1-2\gamma)$.*

Proof. We proceed using backwards induction and employ the same procedure as the proof of Lemma 3. □

With these two lemmas in place we can prove game-theoretic security using similar arguments as before. This allows us to establish the following:

Theorem 4. *With transaction fees of size τ, the contract has $[x(1-2\gamma)-\tau]$-strong game-theoretic security when $\lambda = x$ and $x - x' > \tau$.* □

B. *Denial-of-service attacks*

We briefly consider a malicious sellers that wastes the buyer's time and resources by accepting a contract only for it to time out. When there are no transaction fees, this attack is free to deploy and clearly imposes negative utility on the buyer, since their funds are locked until the timeout passes. With transaction fees, the attack is no longer free, though it is still fairly cheap for large purchases. To circumvent this, we can force the seller to make a deposit in order to accept the contract. The deposit is paid back when contract is completed. In this way, the seller can only mount the attack by suffering a similar utility loss as the buyer which a rational seller would not do.

C. *Coin toss arbitration*

In this section we consider the special case in which the output of the arbiter is independent of the evidence being submitted, i.e. $\gamma = \frac{1}{2}$. The advantage of this is that we can implement such an arbiter using a cryptographic protocol. However, we showed that strong game-theoretic security is only possible when $\gamma < \frac{1}{2}$ so we need to relax our security definition.

Definition 6 (Weak security)**.** *Let π be a protocol with strategy space $\mathcal{S}$ where $s^* \in \mathcal{S}$ is the unique honest strategy profile. We say π enjoys* weak game-theoretic security *if s^* is a subgame perfect equilibrium.*

While this guarantees that being honest is an equilibrium strategy it does not provide a strict incentive to do so. In particular, there is no guarantee that a strategy with weak security is an evolutionary stable strategy. However, it remains secure in a strong sense against risk averse players. Unfortunately, this also means it is strictly insecure against risk seeking players.

Theorem 5. *Using a coin toss arbiter, the contract has weak game-theoretic security for $\gamma = \frac{1}{2}$ and $\lambda = x$.*

Proof. For s^* to be a subgame perfect equilibrium, there must be no $s \neq s^*$ that achieves a strictly larger payoff. As before, this can only be achieved when:

$$x(1-\gamma) - \lambda\gamma \geq x\gamma - \lambda(1-\gamma) \tag{9}$$

which solves to $\lambda = x$ for $\gamma = \frac{1}{2}$. □

We can implement the coin toss arbiter using a variant of Blum's coin flipping protocol [12], [13]. Suppose we have a commitment scheme, and let commit be the commitment function. Then the arbitration proceeds as follows:

1) S samples a random bit $b_S \in_R \{0,1\}$, and a random string $r \in_R \{0,1\}^\kappa$.
2) S computes $C \leftarrow \mathsf{commit}(b,r)$ and submits C to the smart contract.
3) B samples a random bit $b_B \in_R \{0,1\}$ and submits it to the smart contract.
4) S submits b_S and r to the blockchain.
5) The smart contract verifies that b_S, r is a valid opening of C: if not, let $b := 0$. Otherwise let $b := b_S \oplus b_B$.
6) The smart contract transfers $x + \lambda$ to S if and only if $b = 1$, and transfers $x + \lambda$ to B otherwise.

If at some point either party times out, it is assumed they forfeited, and the funds held in escrow are released to the other party. Analysis of the protocol is straight forward: the output is uniform, and security reduces to that of the commitment scheme. From a cryptographic perspective, this protocol is unsatisfactory because it does not satisfy fairness: S can choose not to complete step 4 and simply abort the protocol without revealing the output to B if they are dissatisfied with the result. However, this is not an issue for our application, since S loses the dispute by doing so. In general, it is hard to achieve a fair coin flip on a blockchain: the best known protocol to date samples $\Theta(n^2)$ fair random values using an amortized $O(\log n)$ exponentiations per value [14].

D. *Choice of blockchain*

We did not consider any specific blockchain in the previous sections: our contract works for any blockchain with the capability to execute smart contracts. As a result, the contract inherits many properties of the underlying blockchain, which means the contract can be instantiated in a variety of ways. In this section, we consider some instantiations of the contract in different types of blockchain.

a) Pseudonymity: Instantiating the contract on a public ledger such as Ethereum is the most straight-forward solution. Of course, this means that all transactions are public and available to other parties. However, for some applications this can be considered a feature: having access to the transaction history of a seller indicates how likely they are to cheat and holds the parties somewhat responsible for their actions. Dellarocas [15] shows that under the right conditions, a long-lived seller has strong incentive to behave honestly when faced with many short-term buyers. The incentive is strongest in the initial phase where the seller has to work hard to build up a good reputation and diminishes as their reputation increases.

b) Full anonymity: It is possible to use the contract to facilitate fully anonymous trading by using a blockchain with built-in anonymity (Monero, Zcash, etc.). Doing so necessitates the use of the coin toss arbiter, as it is impossible to remain anonymous when submitting an evidence string containing personal information. However, this makes it impossible to enforce regulation on the goods being transacted, and such a market would likely be used for criminal activity.

c) Revocable anonymity: We can make the contract comply with all laws and regulations by using a blockchain that supports revocable anonymity, e.g. Concordium [16], [17]. To register in such a blockchain, a party needs to identify itself with an identity provider using some formal document. They can then create new anonymous user accounts to be used on the blockchain. Using a designated verifier zero-knowledge protocol, a user can prove to satisfy some predicate on their real identity, such as verifying that their age is ≥ 18. The user are in principle anonymous, but can be *deanonymized* under suitable conditions, say if illegal behavior is suspected. This requires an agreement between several qualified authorities, the so called *anonymity revokers*. For example, the local police, or the local courts may be able to deanonymize users in their relevant jurisdictions. This serves as a "best of both worlds" in that regular users retain their anonymity, while criminal users are subject to legal repercussions.

VII. Implementation in Ethereum

We implemented the smart contract in Solidity and tested it on the Ethereum platform [18]. For simplicity we instantiate the contract using $\alpha = 1$ and the coin-toss arbiter as presented in the last section. The code can be found at (https://github.com/SSODelta/DecentralizedCommerce). To avoid repeatedly covering the large cost of deploying a contract, the seller deploys a single `Vendor` contract that processes all their sales and displays listings. The contract is parameterized by a timeout constant as well as the PGP key of the vendor. A prospective buyer issues a request to the seller's contract by depositing the correct amount of money. The seller can then accept or reject the request. If accepted, the contract proceeds as described in previous sections. To implement the commitment scheme we use the `keccak256` function together with a random nonce. This is secure in the random oracle model. The most expensive part of the interaction is deploying the `Vendor` contract. This fee is paid once by the seller. In our implementation this incurred a transaction cost of $1.4 \cdot 10^6$ gas. With a gas cost of 40 gwei (1ETH = 10^9 gwei), and an exchange rate of 600 USD / ETH, this comes out to approx. 24 USD. Fortunately, this fee only needs to be paid once per vendor. The most expensive recurring part of the transaction is requesting a new purchase from the seller which is paid by the buyer. This costs $1.3 \cdot 10^5$ gas which is approx. 3 USD. All remaining operations cost approx. $5 \cdot 10^4$ gas which is roughly 0.8 USD. For more details, see Table I. This means the (amortized) transaction fee of using the contract for a purchase is approx. 2-3 USD for the seller and 4-5 USD for the buyer. For illustration suppose $\gamma = 0.25$, this gives $[0.5x - 4]$ USD game-theoretic security for the purchase of an item of cost x USD. For $x = 100$ USD, this gives 46 USD security, meaning a dishonest party loses 46 USD (in expectation) by acting dishonestly. The contract can likely be optimized: most of the cost comes from having to store data on the blockchain which is expensive on the Ethereum blockchain. Instead, we can use IPFS to store all the data and only store hashes in Ethereum [19]. This would likely decrease the cost of running the contract. It is worth noting that transaction fees for Ethereum reached record levels in the latter half of 2020. If the same benchmark were done in Dec 2019, the estimates would decrease 15-20-fold.

Seller actions		
Deploying `Vendor` contract	$1.4 \cdot 10^6$ gas	33 USD
Update listing Counter/forfeit dispute	$4.5 \cdot 10^4$ gas	1.1 USD
Accept/reject contract Claim delivery Abort / timeout	$3.6 \cdot 10^4$ gas	0.9 USD
Buyer actions		
Request new purchase	$1.3 \cdot 10^5$ gas	3.1 USD
Issue dispute	$6.0 \cdot 10^4$ gas	1.4 USD
Open commitment	$4.5 \cdot 10^4$ gas	1.1 USD
Confirm delivery Abort / timeout	$3.9 \cdot 10^4$ gas	0.9 USD

TABLE I
Cost of issuing commands in the smart contract by our implementation on the Ethereum blockchain. We assume a gas cost of 40 gwei (1 ETH = 10^9 gwei) and an exhange rate of 600 USD / ETH. It is worth noting that transaction fees have reached record levels in the latter half on 2020; if the contract were benchmarked in Dec 2019 the price would be 15-20x lower.

Acknowledgment

I am grateful to the Aarhus Crypto Group and my advisor Ivan Damgård for supporting this research.

Conclusion

In this paper, we proposed a smart contract for trading any physical goods or services using a smart contract as an escrow. The contract settles disputes by a wager between the buyer and seller, where the parties wager that they can convince an arbiter of their honesty. The contract was shown to be secure in a strong game-theoretic sense for certain values of the size of the wager, assuming the arbiter is biased in favor of honest parties. We showed this was inherent to any such protocol achieving game-theoretic security. By generalizing the contract we were able to obtain tradeoffs between security and size of the wager. We showed how to weaken the security definition to replace the arbiter by replaced by a random coin toss. Finally we implemented the contract in Solidity and evaluated it on the Ethereum blockchain.

Future work: It would be interesting to weaken the security model to dispense with the lower bounds of $\gamma < \frac{1}{2}$ and $\lambda = \Omega(x)$. One option is to introduce a reputation system and take into account the probability with which you get cheated. In order for the contract to be used in practice, there is need for an optimized implementation of the contract with a formal proof of correctness. Finally, empirical studies on arbiters are necessary for parties to obtain estimates of γ.

References

[1] J. Haucap and U. Heimeshoff, "Google, facebook, amazon, ebay: Is the internet driving competition or market monopolization?" *International Economics and Economic Policy*, vol. 11, no. 1-2, pp. 49–61, 2014.

[2] D. S. Dolliver, "Evaluating drug trafficking on the tor network: Silk road 2, the sequel." *The International journal on drug policy*, vol. 26 11, pp. 1113–23, 2015.

[3] S. Dziembowski, L. Eckey, and S. Faust, "Fairswap: How to fairly exchange digital goods," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 967–984.

[4] A. Asgaonkar and B. Krishnamachari, "Solving the buyer and seller's dilemma: A dual-deposit escrow smart contract for provably cheat-proof delivery and payment for a digital good without a trusted mediator," *CoRR*, vol. abs/1806.08379, 2018.

[5] J. Witkowski, S. Seuken, and D. C. Parkes, "Incentive-compatible escrow mechanisms," in *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, ser. AAAI'11. AAAI Press, 2011, p. 751–757.

[6] C. Lesaege, F. Ast, and W. George, "Kleros Short Paper v1.0.7," Tech. Rep., 09 2019.

[7] "Kleros escrow explainer - secure your blockchain transactions today," accessed on 10/10/2020. [Online]. Available: https://blog.kleros.io/kleros-escrow-secure-your-blockchain-transactions-today/

[8] "How moderators and dispute resolution work in openbazaar," accessed on 16/10/2020. [Online]. Available: https://openbazaar.org/blog/how-moderators-and-dispute-resolution-work-in-openbazaar/

[9] M. J. Osborne and A. Rubinstein, *A course in game theory*. Cambridge, USA: The MIT Press, 1994, electronic edition.

[10] I. Abraham, D. Dolev, R. Gonen, and J. Halpern, "Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation," in *Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing*, ser. PODC '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 53–62. [Online]. Available: https://doi.org/10.1145/1146381.1146393

[11] B. Thomas, "On evolutionarily stable sets." Springer Verlag, 1985.

[12] M. Blum, "Coin flipping by telephone a protocol for solving impossible problems," *SIGACT News*, vol. 15, no. 1, p. 23–27, Jan. 1983.

[13] I. Damgård, "Commitment schemes and zero-knowledge protocols," in *Lectures on Data Security, Modern Cryptology in Theory and Practice, Summer School, Aarhus, Denmark, July 1998*. Berlin, Heidelberg: Springer-Verlag, 1999, p. 63–86.

[14] I. Cascudo and B. David, "Albatross: publicly attestable batched randomness based on secret sharing," Cryptology ePrint Archive, Report 2020/644, 2020.

[15] C. Dellarocas, "Reputation mechanisms," in *Handbook on Economics and Information Systems*. Elsevier Publishing, 2006, p. 2006.

[16] I. Damgård, H. Gersbash, U. Maurer, J. B. Nielsen, C. Orlandi, and T. P. Pedersen, "Concordium White Paper, vol. 1.0," Tech. Rep., 04 2020.

[17] J. Camenisch and A. Lysyanskaya, "An efficient system for non-transferable anonymous credentials with optional anonymity revocation," in *Advances in Cryptology — EUROCRYPT 2001*, B. Pfitzmann, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 93–118.

[18] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger."

[19] J. Benet, "Ipfs - content addressed, versioned, p2p file system," 07 2014.