

Decentralized Arbitration Court White Paper v0.1.1

Lesaeger Clément

April 2016

1 Introduction

Smart contracts are smart enough to execute as programmed. However they are not smart enough to depend on elements outside the blockchain or render subjective judgements.

Outside elements can be included inside the blockchain relying on oracles. Those can either be trusted third parties(3) or decentralized organizations relying on game theoretic incentives to have their members submit honest information. The later method is used by the prediction market projects Augur(12) and Gnosis(2). Building a smart contract allowing arbitration by a trusted third party is straightforward and can be done on Bitcoin(11) using multisignature addresses(1). The goal of the Decentralized Arbitration Court is to be a decentralized organisation providing arbitration services while relying on game theoretic incentives to have arbitrators and juries ruling cases correctly.

We cannot simply transpose previous models used for prediction markets because:

- A prediction market oracle output is generally relevant to many parties (5), while an arbitration result is generally relevant to few parties.
- A prediction market oracle output is public, while parties may want to limit the spread of dispute information to the relevant parties (arbitrators and juries).
- A prediction market oracle output tends to have a low level of subjectivity, while arbitration results can sometimes be highly subjective.
- In prediction markets, gathering information tends to be low cost and is a one time process, while in arbitration the cost is higher and involves question and response between the arbitrator and parties. This implies more scalability issues.

Arbitration courts involves new challenges and we aim to solve them in this white paper.

2 Previous Work

2.1 Arbitration through multi-signature addresses

2.2 Schelling-coin mechanism

(7) (14)

3 Project Description

3.1 Network

In order to allow interoperability and take advantage of network the project will be based on Ethereum (6). The goal of the project is not build everything, but rather to be resilient building brick to be used by decentralized applications. We expect all kinds of projects to use the services of the court. For example our system could be used by:

- Simple buying contracts.
- A decentralized Ebay competitor, to handle disputes between buyers and sellers.
- A decentralized Freelancer competitor, to handle disputes between clients and freelancers.
- A decentralized insurance system, to handle disputes rising with insurance claims.

3.2 Arbitrated contract

The Arbitration Court is an opt-in system. This means that smart contract developers must specify how the court can interact with them. They don't need to give them full control over their contracts. In a case of a simple buying contract involving two parties, the contract could allow the court to give the ether belonging to the contract either the buyer or the seller but not allow the court to choose another party to receive the funds.

3.2.1 Hiding plain English contract

In order for an arbitrator or a jury member to arbitrate a contract, he needs to access the plain English (or other language, see the part on subcourts) version of the contract. Storing the plain English version of the contract on the blockchain would have a high gas cost and would cause privacy issues allowing anyone to access it even when the contract does not involve a dispute.

In order to avoid this issue, we use the notion of commitment. A commitment allow parties to commit to a specific information without revealing it during the commitment phase. If needed, a party may reveal it later and prove that the revealed information correspond to the information committed. In our system, both parties will agree on a plain English version of the contract. The party creating the contract will input an hash of the plain English contract. The other party will be able to verify that the hash correspond to the plain English version of the contract before interacting with it. In case of dispute, each party will be able to reveal the plain English version of the contract privately to arbitrators and jury members (by encrypting it using their public keys). Arbitrators and jury members will decrypt the plain English version of the contract and verify that its hash correspond on the one in the smart contract.

In order to avoid the ability to an outside observer to determine which smart contract have the same plain English version, each plain English version of the smart contract must include a random salt.

3.2.2 Requests

The arbitrated smart contract must include functions for parties to make a request to the court. A request would be in the form of an ABI bytecode (code which will determine which function and value of its arguments) that the court will include to a call to the arbitrated contract if the request is

approved. The arbitrated contract must include functions for an opposing party to make a counter-request when a request is submitted to the court. Submitting requests or counter-requests will involve paying an arbitration fee to the court which will only be refunded to the winning party.

If no counter-request is submitted to the court (note that counter-request can simply be doing nothing), after a defined amount of time, the request is automatically granted. If a counter-request is submitted a dispute is created.

3.2.3 Random number created by two opposing parties

First instance arbitrators and juries (when appealed to a jury) are randomly selected. The Ethereum Virtual Machine is deterministic and does not include a resilient random number generator. Using the hash of a block would allow minors to censor blocks with hashes leading to arbiter or juries they don't want to be selected. In order to tackle the issue, a random number will be generated by the two opposing parties.

Again a commitment scheme will be used. The party making a request to the court (the first party) will create locally a random value and submit its hash with the request (the hash is a commitment to the random number). The party making the counter-request (the second party) will submit a random value with its counter-request. After this submission, the first party will reveal its random value which will only be accepted by the court contract if it matches the hash which was submitted. The random value will be computed by applying a bitwise XOR to the random values of both parties.

This scheme produces a random value as long as at least one party provides a random value. The parties could collude to produce any value, thus choosing the arbitrator and the (potential) jury members. However, since those parties are opponents, collusion is not a problematic issue.

Would the first party fail to reveal its committed value after a defined amount of time, the counter-request would be granted by the court.

3.3 Reputation tokens

In order to protect the system from a sybil attack(9), reputation tokens have a fixed supply. They would initially be given to people who have taken part in the crowdfunding of the decentralized court. A lesser part of them will be given to project contributors.

Reputation tokens will determine the probabilities to be drawn as an arbitrator or a jury. They would be valuable because arbitrators and jury members will be granted arbitration fees. Arbitrators and jury members behaving dishonestly will loose part of their tokens while those behaving honestly will win reputation tokens over time.

3.4 Court session

The court works by sessions.

In order to be chosen as an arbitrator, reputation tokens holders will have to activate their tokens for each session of the court. This ensure that arbitrators are only drawn among active members of the court. The probability to be chosen as an arbitrator will be proportional to the amount of tokens activated for arbitration. Since activating a high number of tokens will result in being given an high amount of disputes to arbitrate, reputation tokens holder may choose to only activate part of their tokens to avoid being given more cases than what they can handle. A minimum number of token is set for arbitration in order to avoid drawing arbiters who have "almost nothing to loose" in case of appeal. In order to prevent members from activating more tokens than they have or withdrawing token they could loose, activated tokens cannot be transferred.

Jury members also have to activate their tokens who become non-transferable for the session.

3.5 First Instance Arbitration

Using a jury system in first instance would cause scalability issues and an effect of responsibility dilution among the jury members (most jury members would assume that another member has to investigate the dispute and would not ask extra information to parties themselves). In order to avoid this issue, disputes are handled by arbitrators in first instance.

After the random number of a dispute is created, an arbitrator is randomly drawn. Parties will give him information about the dispute (this includes the plain English contract, see section 3.2.1). The opposing parties will be given a determined amount of time to produce pieces of evidence supporting their case. The arbitrator can ask for additional information. All those exchanges are encrypted using public key cryptography.

After considering all the elements of the dispute (and policies of the court, see section 4.4.1), the arbitrator give a ruling in favor of a party. If the losing party does not oppose the ruling of the arbitrator, the decentralized court contract make a call to the arbitrated contract with the ABI bytecode specified in the request or counter-request. The arbitration fee of the party winning the case is refunded and the arbitration fee of the losing party goes to the arbitrator.

If the losing party believe that the case was ruled improperly, this party can appeal to a jury. Appeals cost extra arbitration fees, therefore parties should appeal only if they believe that the arbitrator behaved dishonestly or made a mistake.

3.6 Jury System

3.6.1 Drawing jury members

The probability to be drawn as a jury member is proportional to the amount of activated tokens. Using the random number created by the opposing parties a segment of the jury activated tokens is drawn. The length of the drawn segment for the first appeal is specified by the court. Token owner can be partially drawn. To show an example, we will assume that the random number for the dispute is $r = 59657401588$, the segment length is $l = 2000$ and we have 6 parties with a total of $t = 10000$ tokens following the repartition shown in figure 4.2.

The randomly drawn segment will be $[r \bmod t, (r + l \bmod t)[= [1588, 3588[$ (assuming we do not wrap around).

Token owner	Activated	Start	End	Drawn
A	1000	1	999	0
B	1500	1000	2499	912
C	500	2500	2999	500
D	3000	3000	5999	588
E	1500	6000	7499	0
F	2500	7500	9999	0

Figure 1: Example of activated token repartition

So owner C will be completely drawn, while owner B and D will only be partially drawn. The number of drawn tokens of an user will be their voting weight.

As with first instance arbitrators, parties will communicate information with jury members. After a defined amount of time, jury member will vote in two step. In the first step they will commit to a vote and in second step they will reveal their vote.

3.6.2 Keeping vote secret during the voting period

In the first step jury members will post a commitment of their vote by posting a hash of their vote concatenated with a secret value. It is important that votes cannot be known before they are revealed, otherwise jury members would be incentivised to vote the same way as previous votes. In order to avoid jury members revealing their vote during the first step, anyone knowing the secret value of user will be able to submit it to the court contract and steal the tokens of this user while invalidating its vote. This won't prevent jury member to tell what they vote for, but this will prevent them to be able to prove their vote early.

In the second step, users will post their secret and their vote to the court. The court contract will verify that the hash correspond to the vote and secret posted by jury members.

3.6.3 Reputation shift

Simple version In order to incentivize honest behaviors, jury members who had voted differently than the outcome or failed to reveal their votes will lose some part (for example $w = 0.1$) of their tokens which will be redistributed to jury members who voted like the majority.

In our example, let's assume that members B and C voted to grant the request while member D voted to grant the counter-request. The request will be granted (unless there is an appeal). Member D will lose 60 tokens, members B (respectively C) will gain 39 (respectively 21) tokens.

If the jury makes the same ruling as the arbitrator, the appeal arbitration fee will be splitted among jury members in proportion of their drawn tokens. If the jury makes a different ruling, the additional arbitration fee will be refunded to the appellant and the arbitrator will lose reputation tokens (having a value equal to the appeal arbitration fee) which will be splitted among jury member in proportion of their drawn tokens. It is important that the value of reputation tokens which can be lost by the arbitrator matches the value of appeal fees in order to avoid jury members having game theoretic incentive toward confirming or overturning the verdict of the first instance arbitrator.

Future version In future versions of the system, we plan to make juries vote whole ballots (including multiple disputes) instead of separated disputes. A machine learning method will be used to weight votes of the juries by a coherence of the vote of a jury with other juries. The method chosen may be an adaptation of the SVD-resolution algorithm described in the Truthcoin white paper (14). Or a method inspired by the work in the field of sensor fusion, as we could consider each jury member to equivalent (on an algorithmic point of view) to a sensor.

Running even simple machine learning algorithms on the blockchain is a challenge due to gas cost and gas limit. In order to tackle this issue, the results of the algorithm would have to be easily disprovable. Then the following scheme is used:

- If there is no result yet, anyone can input the result of the algorithm on the blockchain, putting a deposit.
- If there is already a result, anyone can disprove the current result, put a deposit and put his result.
- After a sufficient time has passed, the party who put the last result can claim all deposits and a small payment for its computation. The last result is considered the result of the algorithm.

This scheme guarantees that once a correct result is input, it will never be replaced (because no one could prove it wrong). It saves on chain computation as we only need the operations to disprove a result to be executable on the blockchain, not necessarily those to find the result. Putting incorrect

results bears negative economic incentives (and those results will be replaced by honest parties proving them wrong), so we expect that most of the time, no wrong result will be input.

In case even disproving a result is too costly in term of gas, interactive proofs could be used(13).

3.6.4 Further appeals

Further appeals double the length of the drawn segment. Cost of appeal is also doubled each time. When the length of the drawn segment is superior to the amount of activated tokens (which means that all potential juries had to rule the case), appeal is not possible anymore.

In case of appeal the result of the final appeal will be used to determine reputation shift. In order to avoid arbitrators and jury members transferring their tokens before the reputation shift, an amount of tokens equal to the amount they could loose is blocked for each user.

4 Improving the court

Those issues will be out of scope for the hackathon. However they would be dealt by the team in the future.

4.1 Changing the court

The court will act as a decentralized organisation. Updates of variables or even of the contract itself could be decided by token holders.

4.2 Voting system

Votes for court improvement and policies will use a different voting mechanism from the one used in disputes. The votes will be held using a liquid voting system (liquid democracy(10) where votes are weighted by the amount of tokens of individuals) where each individual will be able to either vote directly and delegate their vote in a transitive manner. When a individual doesn't votes, his votes are given to his delegate.

For the sake of example, let's assume the following votes, token repartition and delegations.

Token owner	Tokens	Delegate	Vote
A	1000	B	
B	1500	C	
C	500	F	Yes
D	3000	C	Yes
E	1500	F	
F	2500	A	No

Figure 2: Example of activated token repartition

A delegates to B, this makes $1500+1000=2500$ voting rights for B. B delegate to C which, leading C to have $500+2500=3000$ voting rights. Since C votes Yes, the delegation chain stops and 3000 votes are "Yes" votes. D also gives 3000 "Yes" votes which makes a total of $3000+3000=6000$ votes.

E delegates to F which makes F giving $2500+1500=4000$ "No" votes.

The final result is "Yes" (6000 votes against 4000).

In case of mutually exclusive propositions, a Condorcet method will be used. Condorcet methods ensure that if a proposition wins all its duels against other propositions, this proposition is the winning one.

4.3 Sub-court system

The sub-court system allows arbiters and jury members to specialize in particular types of dispute. Token holders can vote for the creation of a sub-court. See section 4.2 for details about the voting system.

For example, from the “General” court, a “B2B” (Business-to-Business), a “B2C” (Business-to-Customer) and a “C2C” (Customer-to-Customer) sub-court could be created. Each of those courts would have different challenges and different policies (see section 4.4.1) to deal with those. B2B would have to deal with information asymmetry, while C2C with lack of reputation from most users.

Sub-courts can also have sub-courts. For example the “B2C” court, could have “B2C Goods” and “B2C Services” sub-courts.

Contracts can opt-in for those specialized courts instead of the general one.

Token holders can activate tokens in at most one sub-court of each court they have token activated.

Using the previous example, Alice can activate his token in the “General”, “B2C” and “B2C Goods” courts. Bob can activate his tokens in “General” and “B2B”. But activating tokens in “B2B” and “B2C Goods” won’t be possible, as token holders need to activate tokens in “B2C” in order to be able to activate tokens in “B2C Goods”.

Disputes in the sub-courts will be handled in a manner similar to to the general one. However, if during an appeal the number of drawn tokens is greater than the number of tokens activated by the court, the dispute will be transferred to the parent court.

This way, contract opting for sub-court will have specialized judges and jury members during the first rounds without lowering the maximum number of jury members who can be drawn in appeal.

4.4 Source of rulings

Determining the correct rulings can be hard, for both arbitrators and juries. And parties submitting their smart contracts to the court may want to know how the disputes will be ruled.

Precedents are considered as guidelines for further rulings, while policies are mandatory.

4.4.1 Policies

Policies are voted by token holders using the process described in section 4.2. Policies are voted by subcourts by all token owner and it is possible for token holder to have a specific delegate for each sub-court. This means that token owners have the right to vote on policies, even in sub-court where they usually don’t activate their tokens. However, we expect most of them not to vote and have delegates which are active in those subcourts.

Text of policies will be stored using Swarm (4). Only a hash of the policies will be stored on the blockchain.

4.4.2 Precedents

After a ruling is given, the arbitrator and jury members will be able to write comments about the case. This comments can contain the main elements of the dispute and the reasoning behind the ruling. Care should be given not include private information.

Those precedents, despite not having any mandatory strength, can be used by arbiters and jury members to help them reasoning on other cases.

The economic incentive of providing comments on disputes is that in case of appeal, the jury members will be able to see those comments which will plead in favor of their rulings (making their ruling more likely to be chosen in case of appeal).

4.5 Disputes involving more than two parties

In case of disputes involving more than two parties, getting a random number becomes a harder challenge.

We will use a scheme similar to RNGesus proposal (8).

4.6 Choice of arbiters

4.7 Lawyer system

4.8 Investigator system

4.9 Mediation system

5 Disclaimer

This is an alpha version of the white paper of the decentralized court project. It was created during the hack.ether hackathon within a limited period of time. Therefore, the design of the court is subject to changes and this paper may contain imprecise or even incorrect statements. If you have any comments or want to be part of the project, you can contact us on the Gitter chat at <https://hack.ether.camp/public/decentralized-court>.

References

- [1] Bitrared. <https://www.bitrated.com/faq/>.
- [2] Gnosis. <https://gnosis.pm/>.
- [3] Oraclize. <http://www.oraclize.it/>.
- [4] Swarm, serverless hosting incentivised peer-to-peer storage and content distribution. <http://swarm-gateways.net/>.
- [5] BUTERIN, V. Decentralized court, post on ethereum reddit. https://www.reddit.com/r/ethereum/comments/4gigydecentralized_court/.
- [6] BUTERIN, V. Ethereum, a next-generation smart contract and decentralized application platform. <https://github.com/ethereum/wiki/wiki/White-Paper>, 2014.
- [7] BUTERIN, V. Schellingcoin: A minimal-trust universal data feed. <https://blog.ethereum.org/2014/03/28/schellingcoin-a-minimal-trust-universal-data-feed/>, 2014.
- [8] CHONG, Z. Rngesus. <http://swarm-gateways.net/>.

- [9] DOUCEUR, J. R. The sybil attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems* (London, UK, UK, 2002), IPTPS '01, Springer-Verlag, pp. 251–260.
- [10] FORD, B. Delegative democracy. <http://www.brynosaurus.com/deleg/deleg.pdf>, 2002.
- [11] NAKAMOTO, S. Bitcoin: A peer-to-peer electronic cash system. <https://bitcoin.org/bitcoin.pdf/>, 2008.
- [12] PETERSON, J., AND KRUG, J. Augur: a decentralized, open-source platform for prediction markets. <http://bravenewcoin.com/assets/Whitepapers/Augur-A-Decentralized-Open-Source-Platform-for-Prediction-Markets.pdf/>, 2015.
- [13] REITWIESSNER, C. From smart contracts to courts with not so smart judges. <https://blog.ethereum.org/2016/02/17/smart-contracts-courts-not-smart-judges/>, 2016.
- [14] SZTORC, P. Truthcoin, peer-to-peer oracle system and prediction marketplace. <http://www.truthcoin.info/papers/truthcoin-whitepaper.pdf/>, 2015.