Forseti

decentralized arbitration/mediation network and reputation system framework based on smart contracts

Disclaimer:
Current version of the document is in active development and can undergo significant changes

Abstract

Forseti is an open / open source protocol that provides a decentralized dispute resolution mechanism and process of transferring reliable data from the real world to the blockchain and their confirmation. In other words, Forseti provides a complete infrastructure for projects that need fair dispute resolution, collection of data from the real world and its validation in the blockchain.

Forseti protocol consists of the following main components:

1.Dispute resolution mechanism (DRM) – an algorithm that randomly, transparently and honestly distributes tasks among arbitrators. Also, DRM is called upon to create incentives for arbitrators to make fair decisions without regard to the opinions of others.

2. Decentralized network of arbitrator / oracle pools (DA/MN) - Self-regulating pools of arbitrators specializing in one field of activity, managed by one of the proposed methods (Token Curated List [1], Liquid Democracy [2], Meritocracy, Direct Democracy). Applications and services built on top of the protocol can access and use public pools of specialists who meet their requirements and create their own pools.
For the resolution of disputes or to provide information arbitrators are rewarded in proportion to their reputation
3.Reputation-based incentive system for arbitrators (ISR)

A reputation-based incentive system designed to allow participants to monetize their earned reputation. Thus, reputation must become one of the most important values within a decentralized network of oracles and arbitrators because it will directly affect the earnings of participants. In this regard, there is a need for a reliable reputation evaluation system. It should be protected from external tampering [3] and will motivate network members to perform their duties honestly.
4. Native Fors protocol token which is used by arbitrators/oracles for protocol management and self-regulation within pools, as well as acting as collateral when creating a pool

Introduction

The emergence of blockchain and smart contracts has opened up opportunities for a more secure and transparent business. The fulfillment of the conditions of the transaction is controlled by the code, and not by the people: as soon as data enters blockchain, smart contracts turn on and fulfill their terms. It is not possible to change, hide or cancel a transaction confirmed by a smart contract. But for today, as a rule, all smart contracts are autonomous within their own respective blockchains and do not interact well with other blocks chains and various information systems. The most common way of solving this problem and transferring information to the block is the use of trusted "oracles" [4].

The problem with this approach is that it forces one to trust a third party, as a consequence leaving room for the rebuttal of the information provided. It is impossible to uncover the full potential of a decentralized economy based on smart contracts and Dapp's until we create an infrastructure to provide data to the blockchain from other information systems, and we create a mechanism for resolving disputes that arise from the work of decentralized applications.

Current issues with blockchain apps.

It is impossible to get external data on the blockchain

Blockchain has a specific format, which is based on complete determinism - the outcome of the processes is completely determined by the algorithm, the values of the input variables and the initial state of the system. Events (transactions) in it are carried out following a strict order, which prevents any modification of the stored data and ensures the security of the system, while limiting its flexibility.

Blockchain is a closed system without access to external data.

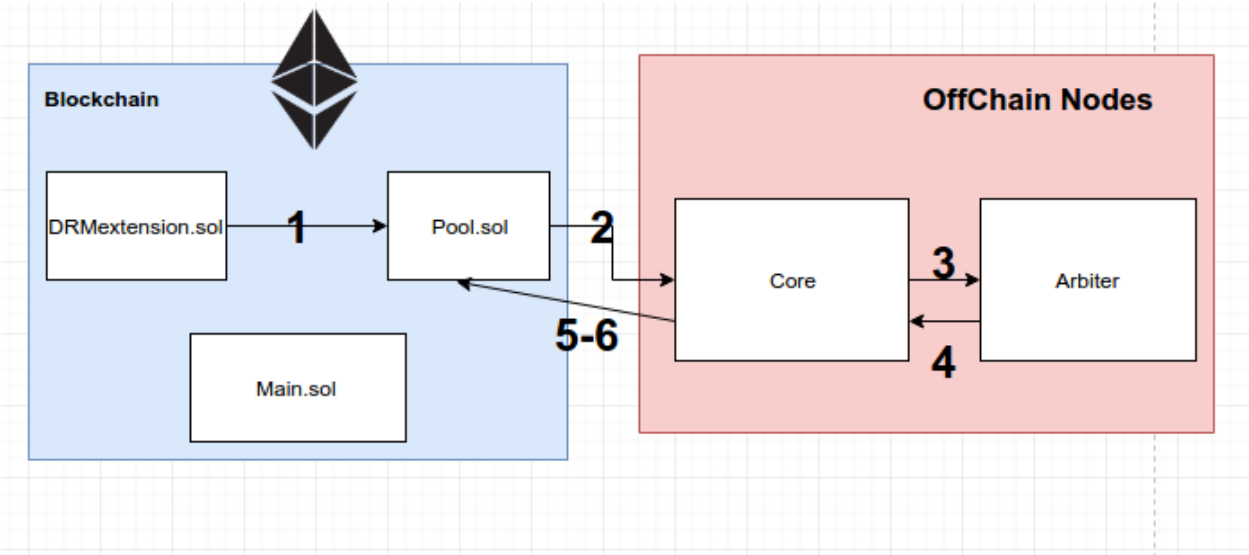
Absence of a dispute resolution mechanism among public blockchains

Due to a lack of trust between the participants, public blockchains need an additional mechanism for dispute resolution that might arise, for example, when conducting transactions with assets that are outside the blockchain. The fact that there is no central trusted authority for dispute resolution in decentralized networks only adds to the difficulty of the situation.

Imperfect oracle system.

Severe need for external information for the work of the blockchain leads to the need for oracles, trusted nodes that can correctly transmit external or subjective data while maintaining its reliability and trustworthiness. The current solution to this problem are oracles [4] that provide data from the outside world. But the current oracles are centralized, and the applications that use them are vulnerable to "single point of failure" attack

Architecture



Our solution can be divided into two principal parts:

Blockchain part

The following elements will be implemented with the use of block technology:

- Pools infrastructure managed by the pool master and participants via a smart contract.
- Data storage with information about all created pools, their participants, as well as pool masters
- Reputation change history of all network members.
- Dispute resolution mechanism (DRM) logic

Offchain part

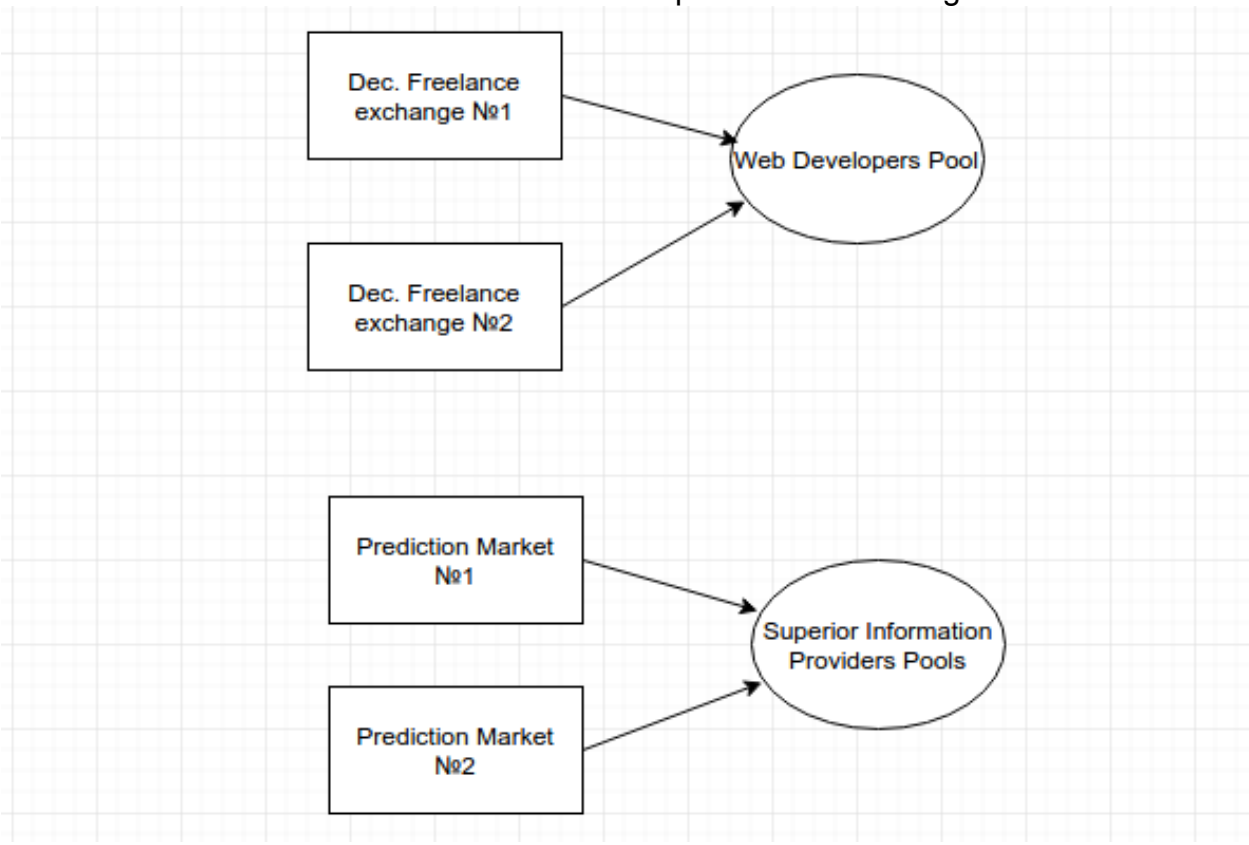
Some functions, mainly the calculations are transferred to the off-chain [5]. This is due to two reasons:

First is the impossibility to carry out all calculations within the main Ethereum network. For example, the distribution of reputation as a result of decay (payment for downtime) will not be possible within the framework of EVM, as rather quickly, with an increase in the number of arbitrators, we will reach the limit in gas block restrictions.

Second is the desire to make our system as easy as possible for potential arbitrators. Since arbitrators in our system may be people who are not familiar with blockchain, for example lawyers, we would like to reduce their interaction with it to a minimum without forcing them to make a transaction for every action.

Decentralized network of arbitrators / oracles pools (DA/MN)

It is difficult or practically impossible to create a unified dispute resolution system, because different tasks imply a different set of skills of the arbitrator, so in our protocol the arbitrators will be divided into pools according to their skills.



Applications and services built on top of the protocol can booth use public pools of specialists and create their own. Joint use of public pools by various services will allow them to solve the problem of liquidity of arbitrators and remove from them the responsibility for the search and management of specialist's pools. The arbitrators receive rewards distributed proportionally to their reputation for the resolution of disputes or the provision of information. The size of the desired reward is set by the pools themselves, depending on their management system.

Creating a pool

In order to create a pool, the pool creator must freeze a certain number of Fors tokens. The number of tokens depends on the estimated size of the pool and the commission of the master for maintaining the pool.

Maintaining (operating) the pool

Maintaining the pool, involves both calculating the distribution of reputation and reputation decay for all pool members, selecting arbitrators when pool gets a dispute, communicating the resolution, and periodically recording (anchoring) current state of the pool to the core network.

The pool master is entitled to receive a percentage of transactions allowed by the pool for maintaining it at the time of recording the current state, and after the expiration of the rebuttal period.

Verifying calculations \ rebuttals

As the calculations made by the master occur outside the main network, there exists a possibility for the pool master to manipulate the reputation data and the voting results. For example, he can choose arbitrators not in accordance with the salt received from the parties to the transaction, and, conspiring with one of the parties, hand pick arbitrators instead of making a random selection. Or the master can deliberately overestimate or underestimate the reputation of the arbitrator when recording them in the main blockchain.

In order to avoid such situations, there is a process of authenticating the data of the master pool. Any arbitrator or master pool may dispute the accuracy of the data entered by the suspected master pool by sending a special "poisonous transaction" [6], following which proceedings will be opened. If the master of the pool is recognized as wrong, he loses the frozen tokens stake and also does not receive a percentage of transactions for the reporting period.

Reputation-based incentive system for arbitrators (ISR)

Forseti reputation protocol is designed to minimize the possibility of "cheating" with reputation points and motivate network members to perform their duties faithfully. Developing our reputation protocol we decided to stick to the following principles:

- Reputation has to be a value for the participants in the protocol
- The possibility of cheating should be minimized
- Pool members should be motivated to continuously participate in the process

Reputation points are a key feature of the protocol. The total number of reputation points is different for each pool and is always equal to the number of participants inside the pool.

The fact that the number of reputation points is fixed provides protection from Sybil attacks [7, 11] and at the same time gives the network an effective way to fine participants.

Reputation protocol also provides for the decay of reputation (payment for downtime) - a process in which pool members lose their accumulated reputation points, if they do not use them to resolve emerging disputes and to process requests for status and changes in other information systems. In this sense, it can be said that reputation points are also an obligation, not just an asset, as their owners are required to use their points, otherwise they will lose them.

Main features of the reputation protocol:

- Each new member of the pool initially has 1 reputation point
- The sum of all reputation points within the pool is always equal to the number of its participants
- Reputation points are earned by oracles/arbitrators when, with their approval, about the task they received according to most other appointed oracles / arbitrators.
- Reputation points are lost by oracles, when their decision is contrary to the decision of the majority. Voices are counted according to the reputation of the voter.
- All members of the network regularly lose reputation points in the form of "forfeit" (decay of reputation), these points are distributed between honest oracles / arbitrators.
- The total number of reputation points received by pool members is always equal to the total number of lost points, including the reputation decay

Reputation decay

Progressive fee (the more the reputation, the more you lose) for downtime or non-participation in the system.
To ensure that the most authoritative members of the network also carry a greater responsibility, the function of the of reputation decay has been modeled in such a way that it applies a deduction for each participant in proportion to his reputation. Thus, the reputation of the most authoritative participants decays more quickly, while the evaluation of the youngest participants remains virtually untouched.

Once in n blocks, all reputation points of all users are burned according to formula

$$Vt=Vt-1Dkln(Vt-1)$$

D and k - parameters responsible for the rate reputation decay. D lies in the range [0,1]; Vt-1- the number of reputation points prior to the application of the reputation decay function.

In this way, the reputation of the most authoritative participants decays more quickly, while the evaluation of the youngest participants remains virtually untouched.

Such a progressive decay model can also be regarded as a measure of the struggle against centralization.

	1	10	20	30	40	50	60	70	80	90
1	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
10	9.771240	9.549933	9.335789	9.128526	8.927879	8.733593	8.545425	8.363141	8.186520	8.015347
20	19.406812	18.836917	18.289236	17.762744	17.256474	16.769507	16.300970	15.850038	15.415925	14.997888
30	28.991834	28.027175	27.103830	26.219731	25.372925	24.561571	23.783929	23.038356	22.323300	21.637293
40	38.544175	37.155177	35.829447	34.563640	33.354607	32.199387	31.095191	30.039394	29.029521	28.063242
50	48.072287	46.237163	44.489493	42.824462	41.237551	39.724517	38.281377	36.904387	35.590028	34.334993
60	57.581137	55.282647	53.097640	51.019669	49.042692	47.161049	45.369436	43.662878	42.036706	40.486540
70	67.073997	64.297887	61.662868	59.160710	56.783721	54.524703	52.376921	50.334068	48.390236	46.539891
80	76.553190	73.287318	70.191523	67.255667	64.470286	61.826537	59.316158	56.931422	54.665100	52.510431
90	86.020451	82.254245	78.688331	75.310546	72.109547	69.074749	66.196273	63.464893	60.871988	58.409500

An example of the impact of the reputation decay function on the participants of the system with different amount of reputation points.

Reputation Points Distribution

The algorithm for distributing reputation points that they receive as a reward for, as well as the breakdown of reputation among pool members, is built in such a way that the reputation does not accumulate in the hands of a small number of oracles, and thus does not create monopolists capable of cardinally changing the outcome of voting.

Our idea is to create a strong "middle class" by allocating the largest number of distributed reputation points to participants with an average reputation score. On the other hand, low-rating participants and, accordingly, high-ranking, will receive much less points with the distribution of reputation points.

This approach is very similar to the normal distribution, which a lot of processes that surround us obey. We have decided to take the formula of normal distribution for distribution of reputation points.

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

Let us study an example:

We assume that 10 members of our pool at the time of distribution of reputation, should receive reputation points. Suppose they have a distribution of reputational points, such as indicated in Figure 1 and we need to divide 1 points of reputation between them.

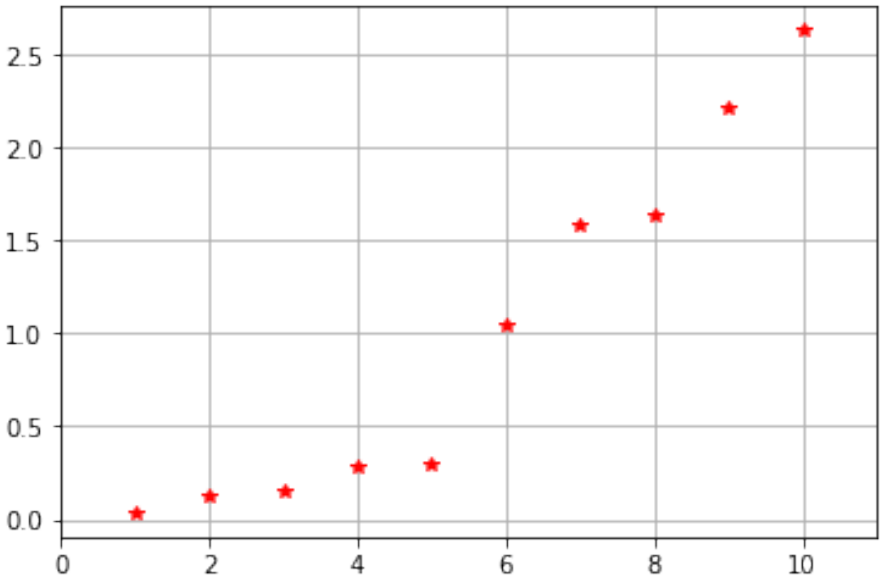


Figure 3. The graph shows the current distribution of reputation points of bona fide arbitrators between whom it is necessary to divide the earned reputation points.

The reputation is then distributed according to the schedule in Figure 2.

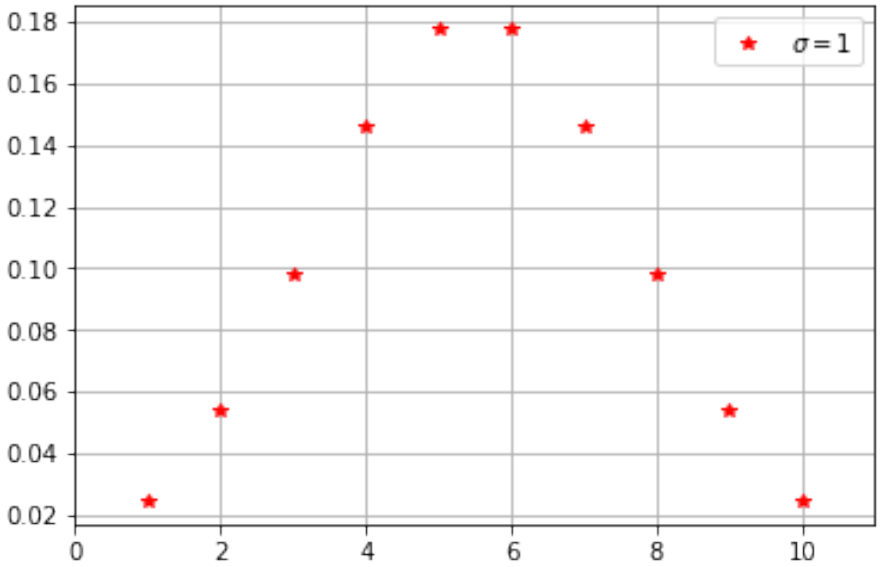


Figure 4. The graph shows the proportion of the total number of reputation points earned, which will be awarded to corresponding arbitrators, see Figure 3.

Accordingly, the lowest and highest rated participants will receive the smallest shares. Participants rated 5 and 6 will receive the biggest shares, according to this case - 0.18 point each. An example with the distribution of 1 reputation point is good so that Figure 2 shows the shares received by network participants in it. It is not difficult to guess that if we were to distribute 2 points, then 5th and 6th participants would receive 0.36 and so on.

Thus, we are trying to strengthen the "middle class", and in this way preventing the Sybil attack [7]

Dispute resolution mechanism (DRM)

Designing our dispute resolution mechanism, we decided to follow two basic principles

1. Modularity
The ability to use our dispute resolution mechanism without significantly changing the logic of the connected decentralized application or smart contract

2. Counteracting decision-making manipulation
The mechanism should create incentives for arbitrators to make honest decisions without regard for the opinions of others, and objectively assess the voice of each of the network participants according to his earned reputation.

DRM Extension

When designing our protocol, we tried to make it as modular as possible, realizing that most services may have specific transaction contract logic or DRM (Dispute Resolution) process that might be different from our DRM implementation. Because of this, the pool does not need to know all the logic of the transaction of the service connected to it, linking our Extension to the service without changing the structure of its contracts is enough.

Arbitration process

Choosing the judges

The process of selecting judges takes place off-chain using the commit-reveal mechanism [8], in order to provide several important properties:

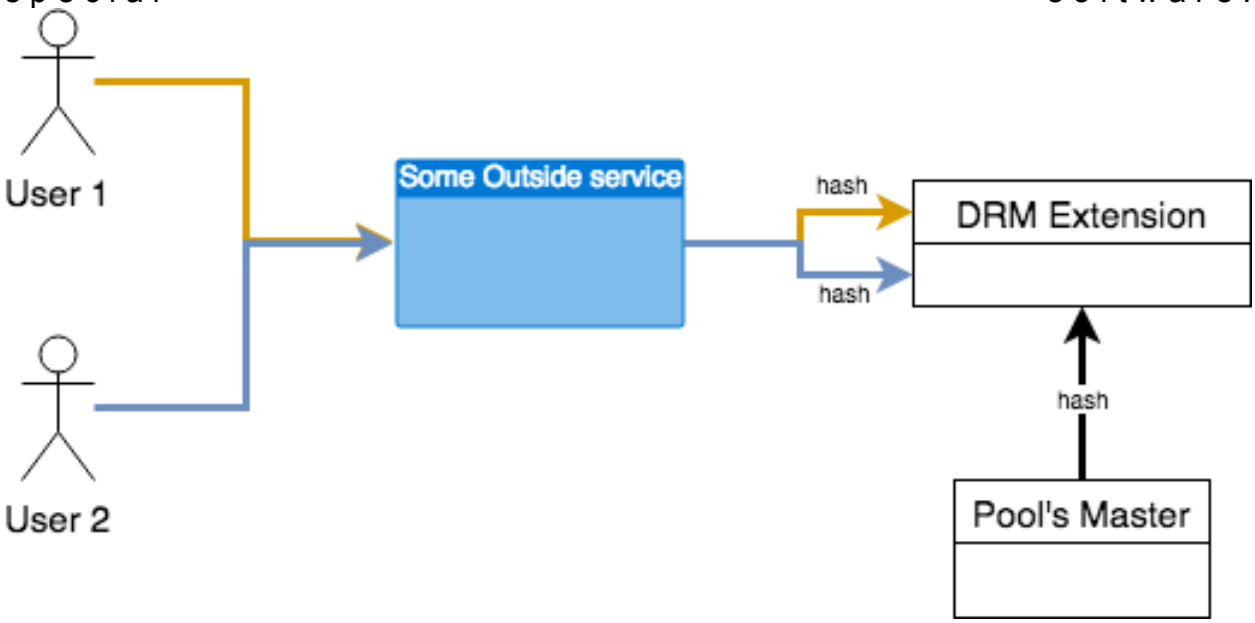
a) Anonymity

Until the dispute is resolved, the parties to the conflict, like the judges themselves, should not know who the judges are in the current dispute. In the case of arbitrators, this is important in order to avoid conspiracy of judges, and the possibility to vote at the last moment, thereby adjusting to the crowd, and not making independent decisions. It is also done to make it more difficult for customers to bribe somebody from the judges, because of the lack of knowledge about them

b) The randomness of the process of selecting judges

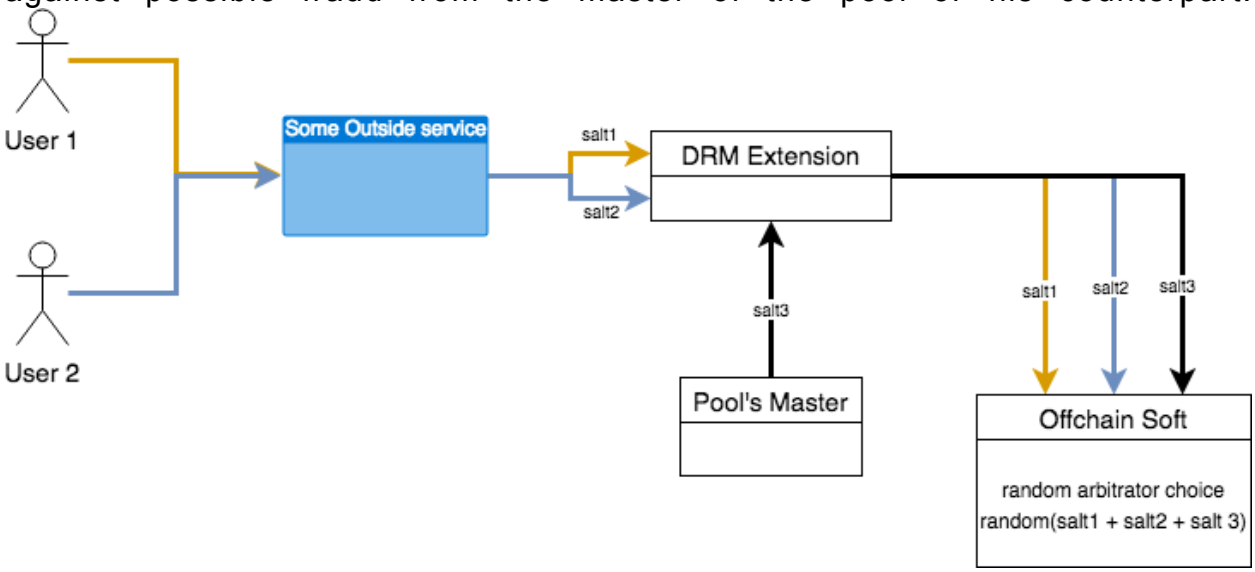
Blockchain is known to be a deterministic system, and it is rather difficult to get a truly random value out of it. Existing approaches using a hash or block timestamp are vulnerable to manipulation [9, 10]. In this regard, the choice of judges in our system is off-chain and is designed according to the following scheme:

When the dispute is initialized, each participant of the transaction sends a random "phrase", which will be used later in the generation of the pseudo-random sequence. But the dispute participants do not do this in an open manner, they send only hashes from their parts of the seed, so none of the participants in the transaction can know the original "phrase." In addition to the participants in the transaction, the encrypted random phrase should also be provided by the master of the pool, using special software.



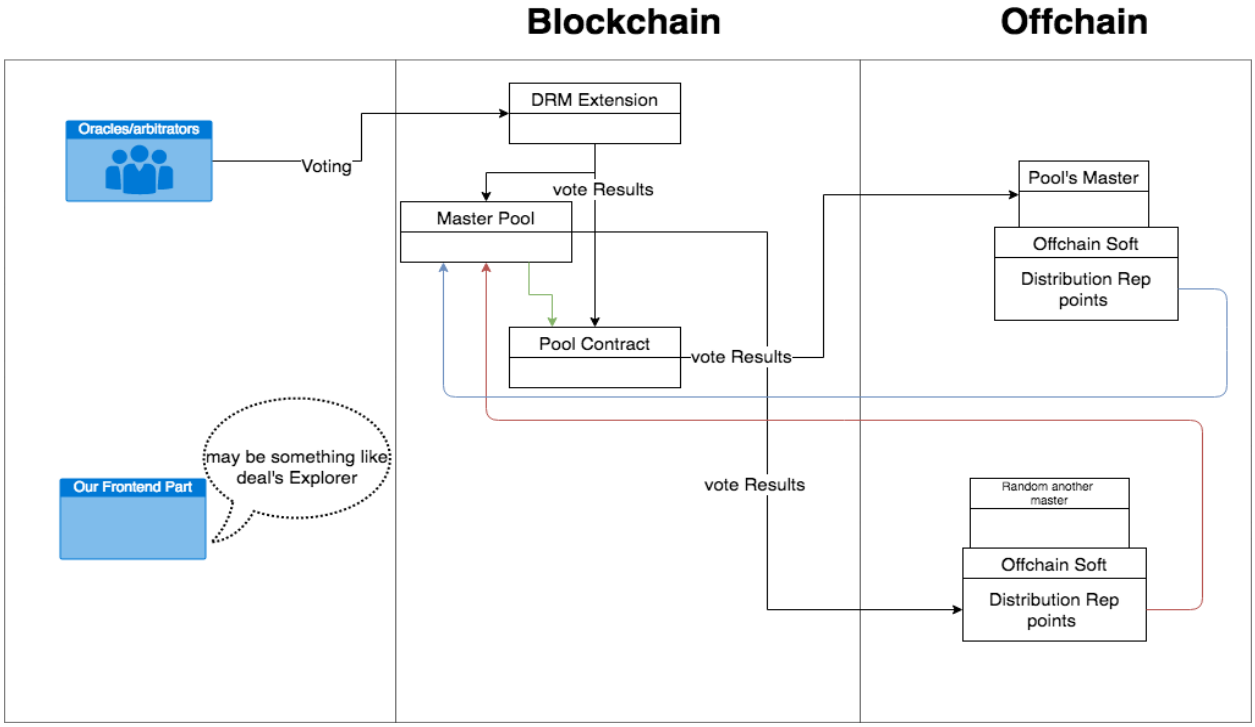
Next, all participants must disclose their pass phrase. In case one of the participants does not do this on time, the seed for generating the pseudo-random sequence is formed without taking into account the participant's phrase, who did not disclose it on time. In this way, each participant in the dispute can protect himself

against possible fraud from the master of the pool or his counterpart.



Judges vote

Voting of judges is also off-chain. Until voting is completed, the current status of the distribution of votes is unknown. The voice of each judge has a different weight, depending on the number of reputation points of the judge. The winning party is the one that has the greatest number of votes cast for it.



Completion of voting

Arbitrators who voted correctly (found themselves in the majority) earn reputation points and a commission from the transaction amount. Arbitrators who have passed the wrong verdict on the contrary lose their reputation points.

Protocol token

Cryptoeconomic protocols create financial incentives for rational economic agents to reach a certain consensus in relation to any process. [11]

Fundamentally, Forseti is a crypto-economic protocol designed to serve as an open standard for decentralized applications that use the mechanism to resolve disputes between the buyer / seller, supplier / consumer of services \ information.

Active participation in the protocol can be optional, however, active participants influence the entire network.

In Forseti, the Fors token has two main roles:

Protocol Management

To improve the efficiency of protocol management, all major decisions will be made by token holders. An example of renewable functions and contracts may be the function of the reputation decay: in case arbitrators entering the pool decide that the penalty is too small, or vice versa, they can change the rate of reputation decay. Renewability is important to the success of the protocol, as it must adapt to changes in services that use it, as well as to changing market requirements.

Pool manager stake

Pool master is rewarded for every transaction that is successfully processed by the members of his pool, and it is profitable for him to keep his pool as demanded as possible. In this connection, he is interested in ensuring that transactions are treated fairly and the number of oracles in his pool increases.

In addition, as stated earlier, before creating the pool, the pool master should freeze a certain number of tokens as a stake. In the event of misconduct by the master of the pool, any arbitrator or other master of the pool may send a "poisonous" transaction indicating this. If the community recognizes the action to be illegal, the pool master will lose frozen tokens, as well as the right to be the master of the pool.

Incentives for system participants

Oracles can monetize the reputation that they earn. They receive a reward for correctly provided information on incoming inquiries and commissions for transactions in which they acted as arbitrators. The size of the commission directly depends on the reputation of the arbitrator / oracle, so it will be profitable for him to have a high level of reputation and strive to raise it. With the growth of the reputation of the arbitrator, the amount of

reward for processed transactions and requests for status increases, as well as do the chances that the arbiter will be included in the next transaction.

Such scheme resembles the cryptocurrencies mining process, where transactions and requests for status are blocks, and reputation is hash rate

References

[1] Token-Curated Registries 1.0 https://medium.com/@ilovebagels/token-curated-registries-1-0-61a232f8dac7
[2] Liquid democracy what that ?! https://medium.com/giveth/liquid-democracy-what-that-bd3c63e8df52
[3] про накрутку
[4] Blockchain Oracles https://blockchainhub.net/blockchain-oracles/
[5] Off-chain transactions https://en.bitcoin.it/wiki/Off-Chain_Transactions
[6] Poison transaction as a Microblock Fork Prevention tool https://www.usenix.org/system/files/conference/nsdi16/nsdi16-paper-eyal.pdf
[7] Sybil attacks https://en.wikipedia.org/wiki/Sybil_attack
[8] Commit-Reveal voting https://karl.tech/learning-solidity-part-2-voting/
[9]Timestamp depedency https://github.com/ethereum/wiki/wiki/Safety#timestamp-dependence
[10] Generating random number on blockchain https://github.com/ethereum/wiki/wiki/Safety#remember-that-on-chain-data-is-public
[11]Attack Classification on Reputations systems https://en.wikipedia.org/wiki/Reputation_system#Attack_classification
[12] Difference between appcoins and protocol tokens https://blog.0xproject.com/the-difference-between-app-coins-and-protocol-tokens-7281a428348c