Randao: Verifiable Random Number Generation

randao.org

September 11, 2017

Abstract

Randao is based on blockchain technology and provides the service of random number generation that is open source, decentralized, socialized and verifiably fair.

Randao not only has the characteristics of uncontrollability and unpredictability which it inherits from the common random number generator, but also it is more accessible and provably fair. ăRandao helps individuals to observe their impact on the generation process of random numbers through providing each stakeholder participation channel. The transparent, irreversible generation process ensures the fairness of the random numbers.

With the Randao service, users can quickly build a verifiably fair application for different scenarios, for example, public management, entertainment, sports, finance, internal management, etc.

Part I
Problem & Objectives

Demand and application for randomness can be seen everywhere in our daily lives such as games, game analysis, sampling, fair distribution and so on. In order to generate random numbers, a variety of methods were invented: the use of dice, roulette wheels, coin-flipping and other statistical methods; the use of the computer languages; the use of quantum mechanic principles and so on. ăThese kinds ofătraditional random generation problems have been given sufficient research and application.

Although the above random number generators perfectly avoid some potential problems on random numbers randomness, uncontrollability and unpredictability, the lack of centralism and verifiable fairness still need to be improved. A true random number is not verifiably fair by nature, here is an example provided by Beacon from NIST (National Institute of Standards and Technology), even if the NIST got the entropy that was sampled from the radiation of universe, they still knew the latest random numbers ahead of others and had the ability to select and interfere with the existing random numbers.

Consequently, people want to find a random number generator and release mechanism with higher fairness. The blockchain plays a role of decentralized platform and provides a natural basis for the verifiable fair random number generation.

However, it is very difficult to design a practicable random number generator based on a public blockchain. In addition to the basic statistical requirements on random numbers, a useful random number generator on the public blockchain should at least to be unpredictable, uncontrollable, unalterable and verifiably fair. Besides, the nature of blockchain is a state replicator, different computing devices share all historical data of the blockchain and keep consistent through Replay. If we want every node to generate random number autonomously during Reply and take that number as the blockchain random number, the random number generator on each and every node should produce the same result. However a fair random number generator requires that no node could possibly know the result in advance, which obviously contradicts this approach. Therefore, the introduction of randomness in the blockchain is different from a single computers implementation.

In the case that there is only one random number generator participant on the blockchain, it is obvious that this participant can know the results of upcoming generated random numbers in advance so the fairness cannot be guaranteed. If it is multiplayer random number generator, there must have a last participant and this last participant has greater power than other participants: to know the generated random numbers in advance; to choose what context to be submitted or even submit nothing depending on the conditions that can influence the final result. Besides, the multiplayer generation make it possible to collude, so the mechanism should be improved to increase the difficulty of colluding and reduce the probability of conspiracy.

In conclusion, how to design a practical and verifiably fair random number generator based on the blockchain has become an important research problem in recent years. Since the Randao team stated the Commit Reveal program in 2005, there were programs Randao++[1] proposed by Vitalik Buterin, some DApps that using Oraclize to obtain the random numbers from the off-chain service to implement a blockchain random number generator. The Ethereum Foundation also listed random number generation on the blockchain as an important issue[2] to be researched for the next 2 to 3 years and invited all parties to participate on a solution.

The paper describes in detail about how Randao using the Commit Reveal, BLS programs to generate and release random numbers to meet the aim of randomness, uncontrollability, multi-participatory and conspiracy-resistant.

Part II
Random Numbers & Random Number Generators

1 True & Pseudo Random Numbers

Random numbers can be divided into two types[3], pseudo random numbers and true random numbers, depending on the method used in generation and its controllability and predictability.

Pseudo random numbers are usually generated by an algorithm, while their distribution functions and correlations can be statistically tested. But in comparison to true random numbers, they are generated by algorithms rather than by a truly random process. Pseudo random numbers try to approach the true randomness as much as possible. Considering that they always have a seed value, they are to some extent predictable. Middle-square Method, LCG, LFSR, and the Mersenne Twister algorithm are widely used in pseudo random number generation.

Most of the programming languages in the world provide users with an easily accessible programming interface or system calls to generate random numbers, like java.Math.Random() and java.util.Random() in Java Language, by which the numbers generated are pseudo random numbers. If anyone gets access to either the seed from which the generation started, or all random numbers generated in a series, one can predict the rest of the numbers that will be generated.

However, in true random number generation, the process is not predictable, neither will two identical true random number sequences be created. In true random number generation, physical phenomena are the general method, such as coin flipping, dice, mouse movement, roulette wheels, electromagnetic noise, atmospheric noise, and nuclear decay radiation, etc. It comes with a higher technical requirement and less productive than pseudo random number generators. In addition, when the entropy contains only limited information, it is not certain that the generator can produce true random numbers. True randomness can be further divided into statistical randomness and quantum randomness. It is widely admitted that random numbers generated by quantum mechanism are

[1] https://www.reddit.com/r/ethereum/comments/4mdkku/could_ethereum_do_this_better_tor_project_is/d3v6djb/

[2] https://github.com/ethereum/research/wiki/Problems#21-on-chain-random-number-generation-63

[3] In Computational Physics, random numbers sometimes are divided into three types, pseudo random, quasi-random and true random.

more truly random than by traditional physical method, due to its internal randomness in quantum mechanism.

Linux kernel provides users with a true random number generator in a statistical way, which generates random numbers using machine noise, sourcing from diverse hardware running speed and user-machine interface speed, say keystroke intervals, mouse movements, time interval for specific interrupts and response speed for block IO request, etc. The site http://random.org/ publicly provides true random numbers generated from atmospheric noise. By listening to the noise generated by quantum fluctuations in the atomic nuclei, scientists at the Australian National University built random number generators and provided them to Internet users.

2 Some representative random number generators

There are different needs in random numbers under different circumstances. Some requires a high level of confidentiality and security in case like generating random numbers for cryptographic key. While in case like dice games, it requires publicly visible random numbers that are not predictable, cannot be tampered with after generation and auditable after the process.

2.1 Centralized random number generators

Before the decentralized public ledger, namely the blockchain, all data are private and centralized. Hence the random number generators are centrally controlled, e.g. NIST Randomness Beacon, and random.org, etc.

2.1.1 NIST generator

NIST Randomness Beacon[4] serves as an implementation of a publicly available random number data source. It has two individual random number generators for commercial use, each equipped with an individual physical entropy source and an SP800-90 proven component. A NIST generator seeks to provide unpredictable, autonomous, and consistent random number source. By unpredictable it means no algorithm can predict the number a NIST generator gives. By autonomous it means resistance to irrelevant interference or to interruption during random number distribution. By consistency it means the exact same sequence when being accessed by a group of users.

2.1.2 random.org

random.org generates random number with atmospheric noise, that is, by recording acoustic waves in the atmosphere and detecting their subtle changes as the entropy source for random number generation. random.org also has mentioned the difference between two types of physical phenomena served as entropy sources - quantum events and chaotic systems. Random number generators based on quantum physics use the fact that subatomic particles appear to behave randomly. There appears to be nothing we know of which causes these events, and they are therefore believed by many to be nondeterministic. In comparison, chaotic systems are those in which tiny changes in the initial conditions can result in dramatic changes of the overall behavior of the system, thus cannot be predictable unless one acquire all information at the initial moment. Both methods can ensure an unpredictable random number generator and random.orgs way with atmospheric noise can be viewed as a chaotic system.

2.1.3 Other physical phenomena

In addition to the use of quantum effects and radio frequency noise, there are methods of atomic decay and even lava lamps. Such programs try to provide true random numbers, targeting more of the usage in scientific research. It is essentially a digital improvement of dice. Since the random number is provided by a single organization, it is still a centralized solution.

[4] https://beacon.nist.gov/home

2.2 Bitcoin Beacon

Bonneau, Clark and Goldfeder[5] from Stanford University etc. proposed that the block data of Bitcoin could be used as a public random source independent of third parties. They analyzed the entropy contained in the initial block and the security of the random number generated from the entropy.

But the issue with generating random numbers with data in Bitcoin Blockchain is consideration on general security. It cannot resist Block Withholding Attacks, which means attacker can take advantage in lottery Apps by bribing miners to discard competitors blocks. The reward for each bitcoin block is defined, so that random numbers generated by this method have a fixed upper security limit that cannot be dynamically adjusted according to the specific situation of the application, resulting in a significant limitation on its applicability. In addition, in Bitcoin Beacon, normal users cannot participate in the block mining. Despite the high cost of conspiracy, miners cannot avoid the suspect from corruption. Hence the program is still not a fair random number generator.

2.3 Algorand

The blockchain consensus algorithm mainly solves two problems - who is responsible for mining the block and what to do when there is a fork. Difficulties usually lie on how to solve the fork, that is, when there are two or more legitimate forks, how to let everyone agree on one chain and continue, which requires a unified standard of judgment. Nakamoto consensus chooses the most expensive fork with the highest cost, and verification of the most expensive chain has to be objective and verifiable, which is, the Proof of Work. It does not mean that there is only one way to solve the fork. Such as the God's dice, where everyone agrees on the fairness of the dice. Whenever there is a fork, the dice can decide on a fork. This consensus can also ensure the normal operation of the blockchain system. Algorand uses Verifiable Random Functions (VRFs) to construct a Common Coin, which acts as a Gods dice. When the nodes of the system cannot reach consensus, the hesitant nodes can quickly react according to the results given by Common Coin. Algorand has used VRFs in multiple cases, including selection of the users to propose the next block and to select members of the committee.

So what is the sortition procedure and its characteristics in VRFs? Imagine, there is a Function F and its difficult to publicly find its pre-image. Now Bob sends a message to Alice requiring her to use her Private Key to calculate $F(s, x) = v$ and take the result as a random number output. As s is invisible publicly, its difficult for Bob to tell whether the output v is Alices honest calculation of $F(s, x) = v$. VRF solved this problem. As VRF requires that Alice, along with the output v, sends to Bob a proof and the public key p corresponding with s, so that Bob can verify if v is the honest result of $F(s, x)$ using p and the proof.

Similar to Bitcoin Beacon, VRFs have a disadvantage that makes them unsuitable for a verifiably fair random number generator.

2.4 Dfinity

Dfinity consensus algorithm is also based on the random number, but the difference with Algorand is that the generation of random numbers is achieved by a group of people using the BLS signature algorithm, which is better in terms of security. In Algorand, the issuance of a VRF is done by one single participant, so the participant can choose not to post the signature against his own will. In Dfinity, signatures are generated by a group of people, and no individual can predict the results, and a single person cannot stop the post of signature.

The BLS used in Dfinity is a threshold signature scheme. After grouping the users, the first round is to generate a random number by one group, then each round later, another group sign the random number generated by the last round as the output of this round, thus no member can predict the signature in advance. The signature process uses the BLS signature mechanism to ensure that no individual can predict the signature results in advance during the signature process and therefore cannot manipulate the random number.

[5] https://eprint.iacr.org/2015/1015.pdf

BLS is a good solution to the problem of Block Withholding Attacks, and the random number generating process cannot be manipulated, predicted, and is difficult to collude, making it an ideal random number generation program.

2.5 DAO

Dice is a traditional way to publicly generate random numbers. One throws, and all stakeholders supervise and endorse the results. But while applied in a wider use case, because of the limited number of witnesses at the spot and because its unauditable later, public random numbers generated by dice or by a random draw are often difficult to convince the public of, and are challenged to be a black box operation or unscientific process.

One of the famous examples of such challenge is the lottery system to determine the order conscription to military service in the US in 1969. Under supervision of journalists, television cameras, government officials and notaries, the conscription officer withdrew a sealed capsule from two rollers containing a date and the number, respectively. This number determines the order in which men born on the date will receive notification of enlistment. For example, if the date drawn out by the conscription officer is April 22, the number is 42, then those who would have their 20-year-old birthday on April 22 will be in the 42nd batch to receive notice of enlistment. The conscription officer will continue to draw out all the date and number combinations until all dates are sorted, which determines the order that the conscripts will conscripted in. The Americans quickly found out that the distribution of this figure is not uniform. Men born in November and December were placed significantly later. Someone ran a Monte Carlo simulation and figured out that the probability of the distribution is only 0.09%. There may be many reasons for this low probability, such as the sealed capsule in the drum mixed uneven, but because the results cannot be audited, it is difficult to remove manipulation as a chance.

In the eyes of the enthusiasts of conspiracy theories, everybody in the world is likely to gang up on him, apart from himself. No matter how theoretically perfect, or how tight in practice a random number generation process is, if he cannot participate in it, the process can be challenged as conspiracy. Participation is the key to crack the distrust.

In traditional organizational structures, because of the limited technology of the organization, delegating to an agent is the common solution. Agents are structural centers by nature, with the agents asymmetrical advantage of information, its fairness can only be amended by systematic restrains on the agent. DAO is a decentralized organizational structure, of which the rules are described by code and enforced by code. Any person can join or leave freely, all participants are equal in the system, which meet our goal for public random number generation design. Therefore, blockchain technology and DAO theory can be used to support and guide the design and implementation of public random number generation.

Randao is based on the block chain technology and DAO theory, with low participation threshold as our guiding principle, to solve the opacity and mistrust in random number generation and in the release process, achieving a random number generation system that can prove its fairness.

3 Comments on random number generators

The output of random numbers generated by a single computer are for its own use, so users only need to consider the inherent quality of the random number, that is, whether to meet the statistical standards, without considering of deception and trust. But in a common random number source service, especially one for multiple participants to use the shared random numbers, the evaluation criteria need to be re-examined.

We can divide stakeholders in public random number generation into two categories, the producer and the consumer. Depending on the number of producers, it can also be further divided into individual production and cooperative production; according to the different use cases, it can also be divided into private use and public use. Because there is no question of trust in a private use case, and it is not in the scope of this white paper. Therefore, this paper mainly discusses the evaluation

criteria of the random number generation scheme in the public sphere, as well as the advantages and disadvantages of each scheme and their use cases.

An ideal public random number generation process should first, be fair - all the parties in the random number generation process are absolutely equal, no one has a comparative advantage; second, be open to the public, including the generated steps, the method used, and the results; and third, ensure that the whole process and the results are auditable, and be able to prove that the generated random numbers have not been tampered with.

As per mentioned in the first chapter of the problem and the target, in order to evaluate a random number generation program, we propose the following criteria:

1. Unpredictable: Unpredictability applies to all participants. Neither producer nor consumer, can predict the possible value of the next random number based on historical data, not even a tiny increase in the success rate on prediction, which meets the Markov Property. In the case of public random numbers, it is also required that no one can raise the probability of prediction based on any public information. For example, in Bitcoin Beacon's program, even if one have all the historical data of the block, the public key of the mining pool, the list of transactions to be packed, etc. he cannot take the advantage of prediction.

2. Conspiracy-resistant: In the process of generating random numbers, if some of the parties join together to exchange their own private information, they cannot affect the process of generating random numbers, change the results of random numbers, or have other comparative advantages, such as getting ahead of others the result of the generated random number.

3. No possibility to know in advance: All parties involved in the random number generation get to know the generated numbers at the same time. No one can know the result ahead of time.

4. Tamper-resistant: Producers of the random number cannot forge a number out, and after the generation of a random number, no one can modify the value.

5. Unselectable: In a production process, the result may contain several random numbers generated. The producers can not withhold any results, or replace one with another one.

6. Unconcealable: Producers cannot refuse to disclose the random number after the random number is generated, which means after being generated, a random number will be open to everyone, and cannot be hidden or withdrawn.

7. Open participation: During a random number generation process, stakeholders can easily participate in the process, and the random number generation program should facilitate and be open to public participation, in order to reduce or eliminate the threshold to participate. No one should be deprived of his right of participation.

8. Auditable: After the production of the random numbers, the whole process should be auditable.

9. Low cost: The cost for random number generation should be as low as possible.

10. Response rate: ăThe random number generation process should be fast enough.

We apply the above criteria to the representative generators for the following evaluation:

	random.org	Bitcoin Beacon	Algorand	Dfinity	DAO
Producer	individual	individual	individual	cooperative	cooperative
Predictability	unpredictable	unpredictable	unpredictable	unpredictable	unpredictable
Know in advance	yes	difficult	difficult	difficult	difficult
Conspiracy	yes	difficult	very difficult	very difficult	very difficult
Tamper	easy	resistant	resistant	resistant	resistant
Selectability	selectable	selectable	selectable	unselectable	unselectable
Concealability	concealable	concealable	concealable	difficult	difficult
Participation	closed	closed	closed	open	open
Auditability	unauditable	unauditable	auditable	auditable	auditable
Cost	low	low	medium	medium	high
Response rate	quick	slow	medium	medium	slow

Part III
Randao protocol design and comments

Randao is an on-chain random number service, with its underlying implementation compatible with different technical solutions. In the Randao project blueprint, there are at least two random number generator schemes. With the deepening of research, technological progress and changes in market demand Randao will further realize other random number generation programs. The current program includes Commit Reveal, and BLS.

1 Commit Reveal Scheme

There are three step in random number generation with Commit Reveal.

1.1 Phase I: Collecting effective $sha3(s)$

All participants willing to take part in the production should send m amount of ETH deposit to Contract C within the specific window period (e.g. 6 block time in Ethereum Blockchain, which is around 72s), along with the $sha3(s)$ of a randomly selected number s.

1.2 Phase II: Collecting effective s

After the first step and within the specific window period of the second step, all producers that successfully submitted their $sha3(s)$ should send to Contract C the selected number s. The Contract C verifies if number s matches the $sha3(s)$ submitted in the first step meets the parameters, and if yes, then the Contract C will save s into the seeds of the Function that will generate random numbers.

1.3 Phase III: Calculate the random number, return deposit and send rewards

After collecting all s_i, apply Function $f(s_1, s_2, ..., s_n)$ as the final random numbers, write them into the storage of Contract C, and return the result to all contracts that required this random sequence.

1.4 Security Constrain Solutions

In order to make sure that the result cannot be manipulated, considering both the security and efficiency, there are the listed extra constraints to the Contract C:

1. In Phase I if two identical $sha3(s)$ were submitted, accept only the first one.

2. In Phase I, there will be a requirement for a minimum number of participants. If the number does not reach the minimum, the production process will be considered failed.

3. If the submitted $sha3(s)$ is accepted by Contract C, participants must submit the corresponding s to the Contract.

   (a) If a participant fail to submit the number s in the window period, the m amount of ETH that were deposited in the first step will not be returned.

   (b) If the Contract failed to receive ALL the s in the second step, the production process is considered failed. After returning the fees that other contracts paid, all deposit collected in the first step will be equally transferred back to participants that successfully send their s in the second step.

2 BLS Scheme

2.1 BLS signature scheme

2.1.1 Theory under BLS signature

In order to introduce you the BLS program, we start with the explanation of what is a BLS signature. A group of N members holds a logic private key S (messages signed by S are marked as SIG, and the public key corresponding to S is marked as P). Each member i in the group holds only a part of the key s_i, and no one has the entire S so that no one can calculate SIG directly. In signing process, each member signs a message with their key s_i resulting in sig_i. Only after having received k (a predefined number) number of signatures will the SIG be calculated.

2.1.2 Characteristics of BLS signature

1. The private key S is decided by the network, while the real value will not appear in any calculation process. Unless all members collude together no one knows S, because if only part of the members produce the private key then those members could collude to know S.

2. No one can predict the result SIG after signed by S, thus cannot manipulate the value of SIG.

2.1.3 Initialization

Before the BLS signature scheme works, you need to run an initialization to generate all the private keys for team members and the public key for the whole team. The process for initialization is as follows[6]:

1. Each member i generates his own random sequence ran_i (they keep it confidential from others, including team members).

2. Calculate $send_{ij} = f(ran_i, j)$ with a given function f by BLS, and send the result $send_{ij}$ to member j (every $send_{ij}$ are different corresponding to j; f is a one-way function, which means you cannot calculate ran_i from $send_{ij}$ and j). Member i needs to send the $send_{ij}$ to all member $j(= 1, 2, ..., N)$, when $i = j$, the result $send_{ij}$ will be kept locally.

3. All member j collect the result $send_{1j}, send_{2j}, ..., send_{Nj}$, and verify $send_{ij}(i = 1, 2, ..., N)$ with function v given by BLS. If $v(send_{ij}) = 1$, the result is legitimate, if $v(send_{ij}) = 0$ then not.

4. After verifying all results, use them to calculate your own private key s_j, the corresponding public key pub_j, and the public key P to S.

5. Broadcast pub_j and P to the network.

[6]Distributed Key Generation: https://en.wikipedia.org/wiki/Distributed_key_generation

2.1.4 Execution Steps[7]

1. Each member i signs the message M with private key s_i to get sig_i, and broadcast sig_i.
2. Each member j collects the sig_i that other members broadcasted and verify it with pub_i. If they correspond, each member j accepts it as an effective signature. After having collected k numbers of effective signatures, including your own sig_j, calculate the final signature SIG and broadcast SIG to the network.

2.2 Random number generator using BLS signature

The basic idea to generate random number with BLS signature is that, all nodes in the network are divided into h groups, the first group generates a random number SIG_1, and choose the second group with SIG_1; the second group sign SIG_1 and get SIG_2, and choose the third group with SIG_2; repeat this process till the h group generates SIG_h as the final random number output, and no group will take part in the generation twice. All steps are as follows:

1. All nodes in the network are divided into h groups, $G = g_1, g_2, ..., g_h$.
2. g_1 generates a random number SIG_1 and broadcast it.ăg$_1$ is removed from G.
3. For $i = 2, 3, ..., h$, execute: take $g_{work} = G\left(sig_{i-1} mod |G|\right)$, remove g_{work} from G, sign SIG_{i-1} with g_{work} to get SIG_i, and broadcast it.
4. The final SIG_h is the output random number.

The above process ensures that each step of the signature is a random number, and no group can predict the subsequent groups signature results. Only the last group has the advantage of manipulating SIG_h, and when the last group has no more than k malicious members, SIG_h is safe. At the same time, it is not predictable which group will do the SIG_h generation task.

2.3 Security discussion on BLS

Issue that might happen in the previous execution process:

1. In the initialization of BLS signature, members from one group colluding will make S disclosed, which means group members can know the result from signature in advance.ăTo make this happen, all group members have to share their private key s_i to other members, or they cant get S, thus there is no way to calculate SIG in advance. However, apart from this route, there is an easier way to attack: k number of members collude, generate their private key with BLS signature for these k members and share between them, which will mean that they exclude the other $N - k$ members in the signature and take over the group.
2. In the second step of signature process, if a member collected $k1$ signatures from other members and use the $k1$ signatures and his own signature (before broadcast) to calculate SIG and find out the SIG is not favorable to himself, he can choose to not broadcast his signature. This can cause that only $k1$ signatures are broadcasted in the network, thus resulting in failure of the signature process. When it is possible for more than Nk members to do this, the above situation can happen.
3. Assuming when $k < N/2$, if the whole network has k members that signed a message that is not approved by the majority of the members of the whole network, the k members can "forge" the collective will.
4. If the random number generated is of direct interest to the majority of members, then in the last step in generating the random number it is likely that more than half of the members refuse to give the correct signature SIG_h so that the random number generation would fail.

[7] https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing

To sum up the above 1, 2, and 3, the BLS system can resist $max(k-1, N-k)$ malicious members. In DFINITY, $N = 400$, $k = 201$, only when more than half of the members in a group are malicious members, will the signature failure or forged signature situation happen. In fact, when grouping all nodes in the entire network, the BLS signature method uses the VRF to randomly group members, thus can provide a high fault tolerance if the grouping process is completely honest. Lets say there are $10,000$ users in the entire network, in the case of $3,000$ members being malicious users, the probability that one group will have more than half malicious nodes is less than $1e^{-17}$, in the case of $4,000$ malicious members, the probability is $1e^{-5}$, and the probability reduces with the increase of N ($k = N/2 + 1$). But no matter how high N is, it cannot meet 50% fault tolerance. When the malicious users reach about 50%, the probability will rise rapidly to around 50%. Therefore, it can be seen that this random number generation mechanism is not suitable for the problems directly related to the interests of the whole network. For the problems related only to the interests of some nodes ($< 50\%$) of the whole network, the above four issues will unlikely happen.

2.4 To prevent conspiracy and bribe

2.4.1 In the initialization

- Individuals that are interested in participating in the production using the BLS signature can sign up to the system and the system determines the final participants.
- Set the Feldman secret sharing parameter, and form a BLS service provision group through the distributed key distribution mechanism.
- All members in the BLS service provision group transfer a deposit into the Ethereum address corresponding to the private key they receive. It is prohibited to withdraw this deposit during BLS service period. If anyone tries to drop out from the service before the default period ends, a part of the deposit will be deducted.

2.4.2 During the service

- Customers that want to use the random number service send a request to the smart contract R, including the message to be signed (if N/A, send the hash from actual block) and the fee to be paid. Smart contract R will save the message.
- After the BLS service detects the request for random number generation on the blockchain, members of the off-chain service team sign and combine all the signatures to get the random number, which will be written into the smart contract R by a determined user.

2.4.3 Auditing supervision

- To prevent consumers from colluding with some of the service members, from obtaining some of the contents of the signature results in advance, and providing BLS services with their preferred contents to be signed, and gaining comparative advantages to benefit, we can avoid conspiracy with the following disciplinary regulations, and make the service members not dare to accept bribes to provide private signatures.
- If someone presents the result to a members signature of the content s and the content s does not appear in the signed history of the smart contract, the member will be judged to be illegitimately signed, and fined their deposit.
- Taking into account the possibility of blockchain reorganization, the member may appeal, and their argument should include the block that contains s.
- Members must have a certain deposit in the system, and whoever owns the private key of the deposit will be able to transfer the deposit, which can prevent members from disclosing their own private key to a specific group of people.

- In order to prevent members from transferring the deposit privately before the BLS service period expires, we added a rule, if a member withdraws their deposit in advance of the default expiry, 20% will be deducted from their deposit to other members who are involved and dont withdraw before the default expiry.
- Each BLS service group will have a good-behavior indicator, which consists of deposit status, signature response speed, online rate of nodes, etc. The good-behavior indicator is shown publicly, so that consumers can select from which group they want their service to be done by. The indicator of the service group may affect the fees of its services.

3 Comments on Commit Reveal and BLS Scheme

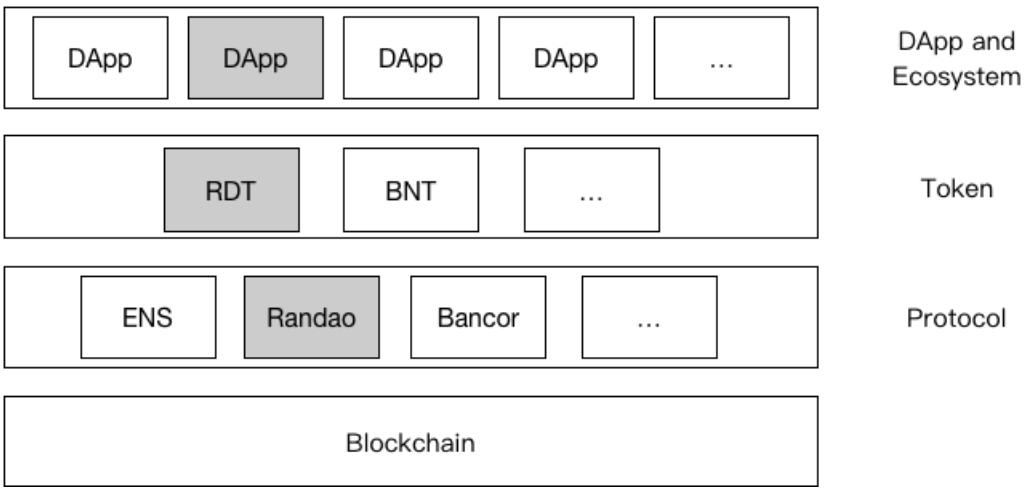
The Commit Reveal Program has the disadvantage of low production in random number generation. It takes at least 10 block time from receiving the production request to the output of a random number. In the case of the Ethereum blockchain at current block time, it takes more than 3 minutes. In addition, participants need to send transactions and submit data several times, thus resulting in a relatively high cost of use. However, the advantage lies in its zero threshold, as anyone can participate in the generation process at any time, which also helps to avoid collusion and to prove its fairness.

The BLS signature scheme is a good complement to the Commit Reveal Program because the generation process is organized off-chain, with a fast response, usually requiring only one block time to generate a random number; the consumer initiates a request for random number generation, the producer writes the random number in the next block, which means it only needs to send two transactions to complete the generation and call of the random number. The production and usage costs are low, suitable for high frequency, and a less harsh demand to be conspiracy-resistance.

Part IV
Randao Technology Architecture

1 Overview

Randao's overall structure is on the blockchain and is committed to a standard and verifiably fair random number protocol on the blockchain. All DApps that are deployed on blockchain and have a demand for random numbers can provide their users with a variety of services based on random numbers through Randao. At the same time, combining with economically incentivized tokens provided by Randao, producers ofărandom numbers can benefit from participating in the process. In addition, around the random number protocol, Randao will also promote the establishment and development of Randao's ecosystem through various forms of technology. The following graph shows Randaos the overall structure on Ethereum:



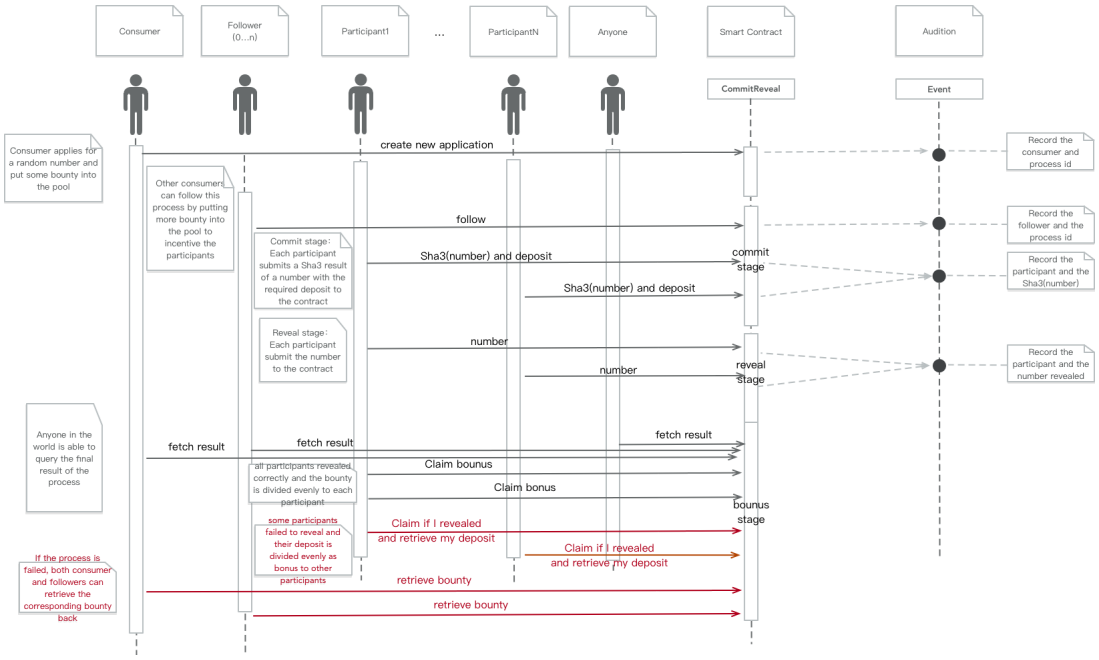
As the graph shows, the protocol layer contains the smart contract that supports the Randao service; the token layer is the implementation of Randaos economic system; and the application and ecosystem layer include the necessary DApps for Randaos operation and marketing.

2 Protocol Layer - Random number protocol

This part is the core of Randao, which will encapsulate various random number generation algorithms and corresponding business logic in the form of smart contracts on one or more blockchains. The diversity of these random number services ensures that our protocol can cover a wider range of business scenarios to meet the needs of more DApps. For example, Commit Reveal is a multi-person participation and multi-stage operation for the random number generation processes, with relatively poor timeliness, while BLS scheme provides a rapid mechanism for random number generation, fitting better the business demand for fast access to random numbers. With the development and improvement of the platform, Randao will combine the market demand to encapsulate more random number generation patterns, to commit Randao to the standard of random number service on the blockchain. Random number services will undergo a rigorous testing and security audit to ensure the quality of its online service.

2.1 Commit Reveal Contract

Commit Reveal is a process that allows multiple people to participate in the generation of random numbers within a specified time. The following sequence diagram shows the process of processing the smart contract:

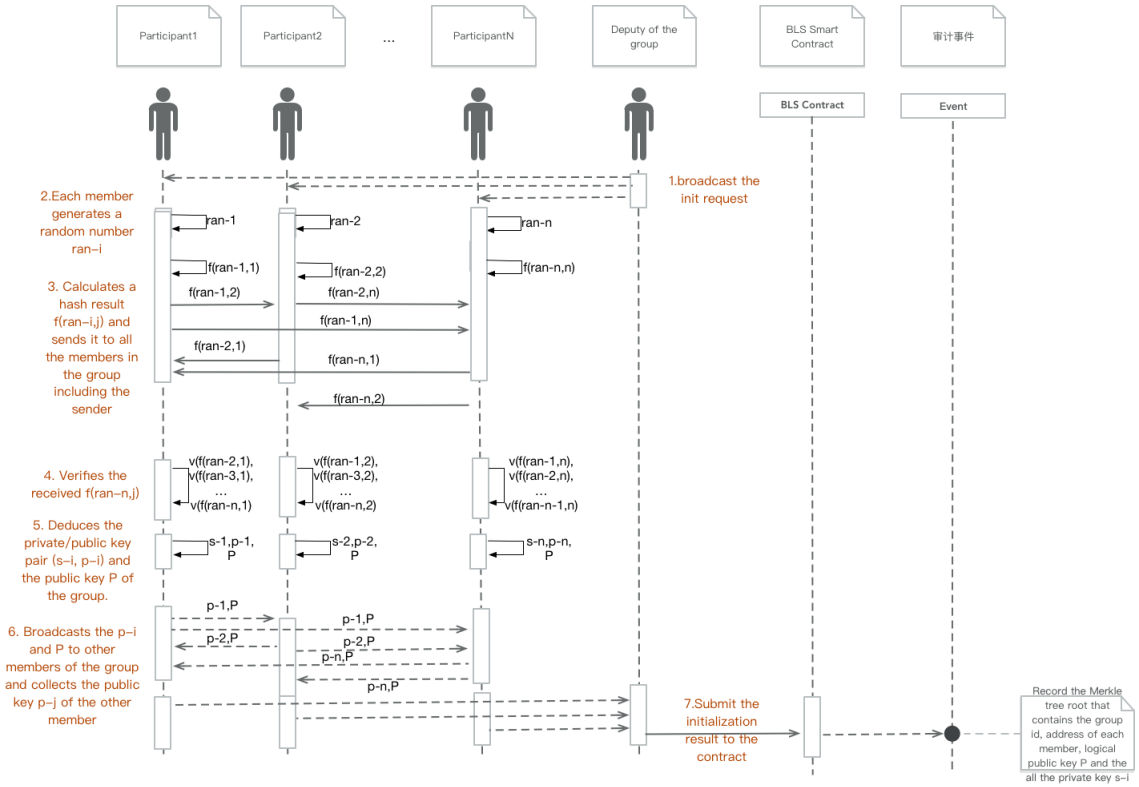


2.2 BLS Contract

BLS is also a method for people to participate in random number generation, aiming to provide a random number service to consumers premised on being provably fair in a more efficient way. First, the BLS scheme involves grouping and initialization. Grouping is to randomize all participants into M groups, each group of N people. Initialization is at the end of the grouping, to generate each members own private key and the public key P for the whole group, according to the BLS signature mechanism. In order to improve efficiency, grouping is preceded off-chain, and the grouping results will run for some time, and then be regularly re-grouped to ensure fairness. Below we describe the BLS grouping initialization and random number generation process through the timing diagram, respectively.

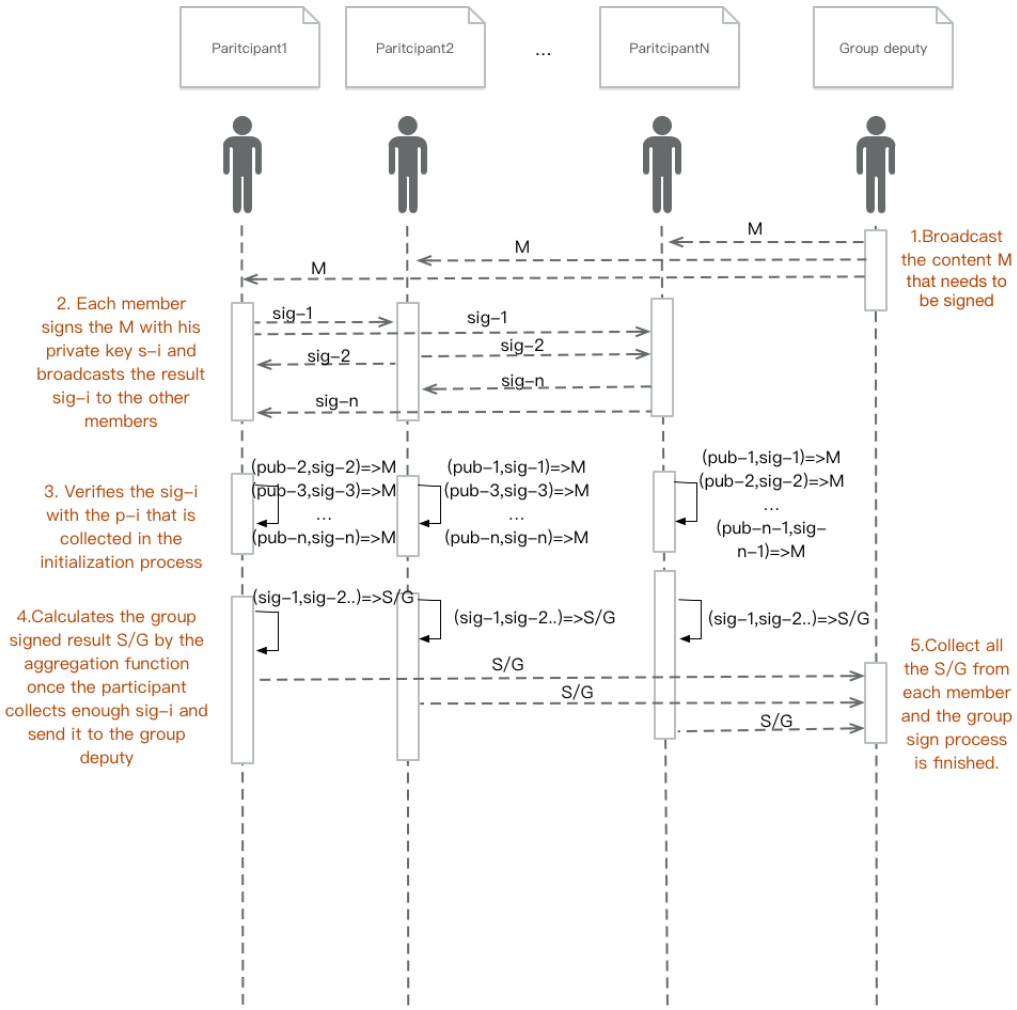
2.2.1 Grouping initialization

Each group consists of members and group representatives, in which the group representative is responsible for transmission and recording of the content and information, not interfering with the random number generation process. Members are those who actually produce random numbers, thus also known as producers. The following graph describes the process of initializing a certificate after the grouping:



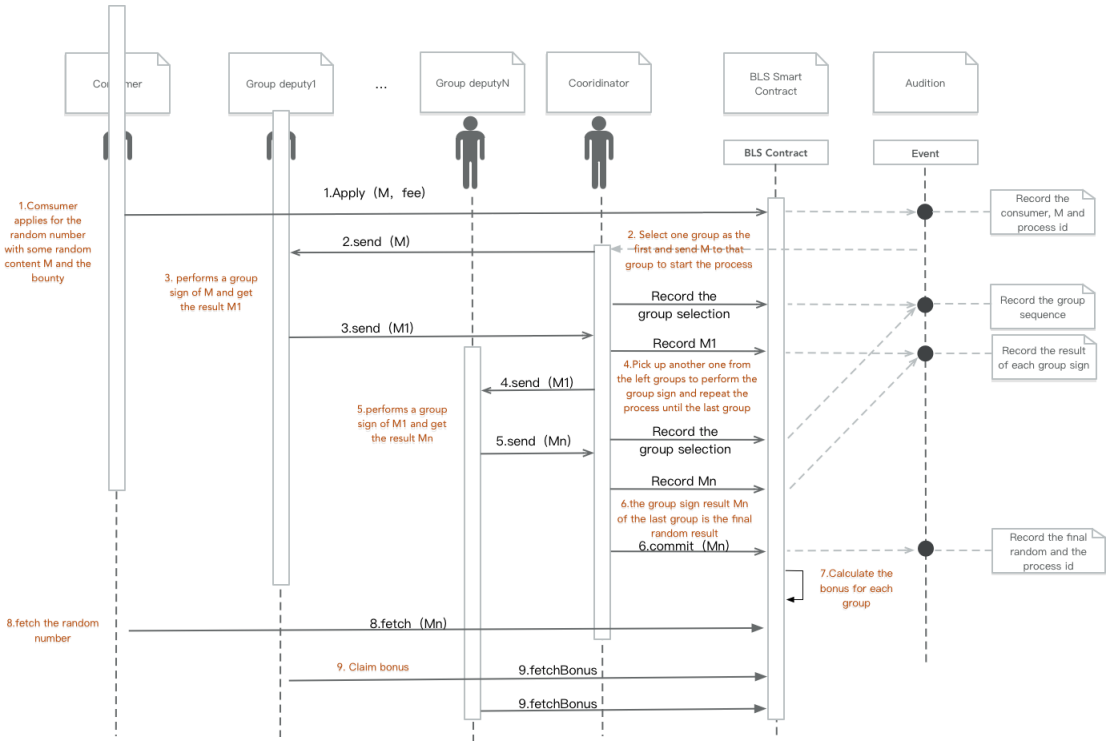
2.2.2 Random content signing process of each group

After the grouping, when receiving a request for this group to sign a specified content, the group members will follow graph to complete the signature process:



2.2.3 BLS random number generation process

After understanding how the group is initialized and signed, the following figure will describe how multiple groups together complete the generation of random numbers:



3 Token Layer - Token

Randao token meets Ethereum ERC20 standard, and is compatible with various wallet apps on the Ethereum Blockchain.

4 Application and Ecosystem - random number access service ecosystem

There is a wide range of demands in the real world for random number generation services. As mentioned earlier, there are a number of centralized service providers, such as random.org, NITS Random Beacon, which provide random number generation services on the Internet. They meet the needs of different scenarios through various types of access services, thus forming a service ecosystem. With the development of blockchain technology, there will be more users and third-party businesses using Randao's decentralized random number protocol. In order to give users an excellent experience, Randao will provide different types of access to support different types of users to create this service ecosystem.

4.1 categories of users

Currently, Randao mainly classify users by three dimensions - their role, their professional level of use, and their user environment:

4.1.1 By role

- Producer: Producers are the users that take part in the random number generation process. An example will be the users that submit the final random numbers in Commit Reveal Program.
- Consumer: Consumers are the users that request random number service from Randao and those who will use the produced random numbers.

4.1.2 By professional level of user

- Dev: Developers refers to the users that provide their own specific service based on Randaos service, they are usually a third-party service provider with access to Randao.
- Normal users: These users dont care as much the technical detail, and wish to directly use Randaos ready product or service.

4.1.3 By user environment

- On-chain user: They are the users that use Randao in the Ethereum Ecosystem.
- Off-chain user: This type of user is usually an entity that has been running a centralized random number service not on a blockchain. For some reason, they dont want to reestablish their business on Ethereum (i.e., do not want to be a professional user in the blockchain environment), but would like to use Randao services on the traditional Internet.

4.2 Construction of the Ecosystem

4.2.1 Ecosystem construction for developers

Developers, either producers or consumers, are based on Randao's services to develop their own products. For such purpose, Randao will create and maintain a developer community, users must be registered in this community as a certified developer to access a variety of API and other development resources provided by Randao. This community contains the following products:

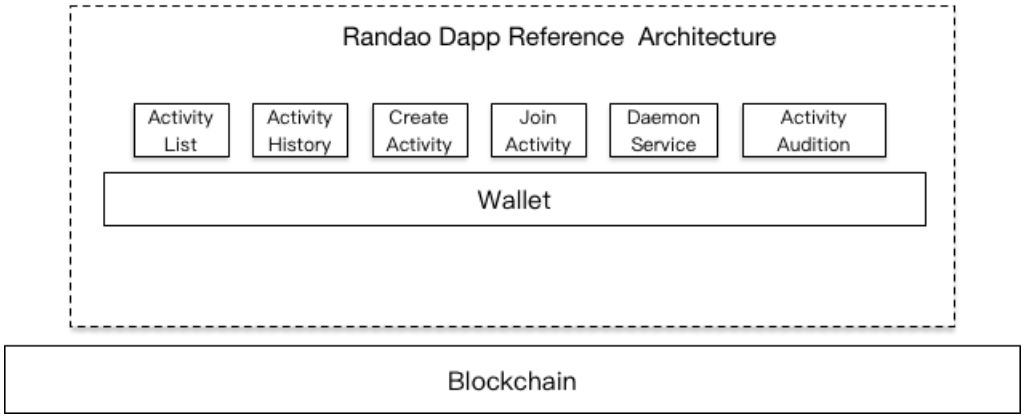
- Detailed Online API Document: which provides Randao API specification and use examples on-chain and off-chain.
- SDK: a SDK for developers to quickly develop an access to Randao service. For an on-chain environment, it is a template based on Solidity to call the Randao contract; while for off-chain environments, Randao provides bridging services, which pack the Randao interface into Restful API, available in Development libraries with multiple languages to facilitate developers to call these APIs.
- Community: to provide developers with online technical Q&A and Randao service discussion, to promote the development and improvement of Randao technology.

4.2.2 Ecosystem construction for normal users

General users do not use Randao for secondary development, but rather use the services directly or indirectly in an intuitive way. Thus, the Randao algorithm will be encapsulated into a DApp or daemon process, where the user can use the following core services and functions:

- Take part in an ongoing generation process as consumer, pay the fee and get the random number output.
- Initiate a random number generation process as consumer, pay the fee and get the random number output.
- Take part in the generation process and get paid.
- Start the producer robot daemon, authorize the system to automatically subscribe to the new request for random number service, once there is a new request, the system will automatically participate in the random number generation service for the user, without users intervention.
- Browse user's participation history in the event.
- Audit the specific data of a finished event.

The picture below shows the reference architecture for Randao DApp:



Part V
Rando Application Scenarios

Using random numbers (sortition) to distribute social resources has been applied to all aspects of daily life. Early from the kindergarten admission qualifications, to the distribution system of high schools, then to the qualifications of buying a car or house, to bidding on government projects, all these decisions depend on sortition. Actually, random numbers help to make many big decisions during peoples lives.

Following are some situations that need random numbers:

- When credibility is needed, i.e. all participants need to agree that the result is fair even if someone doesnt get what they wanted.
- Requirement for the number of participants (more than two participants) is met. The number of participants will not affect the fairness of the results.
- It wont take a long period of time to produce results.

Sortition has been improved to meet the above requirements. In the beginning people needed to get together and use dice, random draw, or similar methods to obtain random numbers, then with the application of the Internet, computers could produce random numbers directly. There have been a series of improvements in the way of participation, production methods and the release methods of random numbers.

However, there is no perfect solution to ensure fairness during the operation process. Dicing, in the early stages of random number generation, was very visual due to its simple and clear operation process and small number of participants. People could confirm the result immediately. Thus flaws like the possibility of black box operations could not influence its credibility a lot. But now, because of the large number of participants it is not possible for each participant to witness the random number generation process. At the same time, the computer programs generally produce pseudo-random numbers depending on the seed, which also creates a higher possibility of black box operations. With the deepening of the publics understanding about random numbers, the above production method is constantly being questioned. In some extreme conditions, it may lead to the loss of credibility of responsible lottery operators.

Randao can be widely used in the above situations and help the responsible lottery operators to rebuild their credibility as a verifiably fair random number generation service. Following this there will be a detailed description of how the Randao service can be applied to a license-plate lottery and random tax audit.

1 License-plate lottery

By the beginning of 2017, there were 2,783,966 participants in the passenger cars license-plate lottery in Beijing, while the effective index is only 13,905 which means the success ratio is less than 1:200. Many lottery strategies online such as non-resident strategies, late participant strategies, and bribing can increase the possibility of success. ăWhether true or false of those rumors, they all come from the peoples doubt and dissatisfaction with the lottery process and its results, besides the fairness of lottery cannot be proved. Every time the result of the lottery is released, people will share their result in their WeChat groups. It is certainly probable that participants will not get it for 6 years, but for normal people they cannot tell if its because of bad luck or other people cheating, they have no way of confirming its fairness. Complaints starts to spread, and spread for a long time, with the spread of rumors the credibility of the lottery operators will also be reduced. ăă ă As a result, under this kind of random number application situation, the ability to verify its fairness and let all participants confirm the results by themselves is very important. Random number generation process needs not only a simple result, but also theăability to communicate, and to meet consensus with participants, which should be the nature of a fair lottery.

Randao is a provably fair service, due to its strong communication characteristics, it can help participants to reach a consensus more efficiently and to eliminate misunderstandings. All participants can submit their own random number through the system and observe the whole generation process. To operate a lottery using Randaos design implements these key points: 1. those people who want to have some influence on the results can choose to participate in the random number generation process; 2. participants should fulfill their obligation and offenders will be punished; 3. the process is transparent; 4. the results can be audited.

2 Random Tax Audit

If people want to get lucky in lottery, they definitely dont want to be luckily picked and checked by authorities if they are cheating the law. In this kind of case, the result will only influence the participants who would be punished, and they wont be able to confirm the fairness of why they were selected to be audited and that topic will not be open to discussion. There is a huge space for the black box operations.

In the case of auditing, the random sampling process is quite casual. For example, the local tax office need to get the sample from more than ten thousand enterprises every year. What really matters is which enterprises can avoid inspection. Because of the internal system of the tax office, the sampling results will not be publicly released, so many agencies claim that they can change the sampling result to help enterprises avoid inspection. The opacity and asymmetry of the information causes many enterprises to rely on luck and take any chance they can to avoid inspection, which directly leads to the loss of the governments credibility. Under this situation, using the provably fair Randao service in their design ensures the fairness of the process and the result. There are many similar situations, for example, customs auditing, hygiene auditing, food safety auditing, and so on. Randaos service can completely eradicate the possibility of benefiting themselves, or harming others, by increasing or decreasing the likelihood of a random audit on themselves or others, and reducing the overall management cost.

Part VI
Randao Economic Analysis

A random number is widely applied in cryptography, numerical calculation simulations, statistical studies, the lottery industry and raffle draws, providing a high business value. In view of the different demands of these situations generation rates, usage frequencies, quality of randomness and the level of random number security, Randao can offer differentiated random number services according to

its algorithmic features. The business value of Randao can be considered from two perspectives. On one hand, Randao, as the verifiably fair provider of a random number generation service, can obtain revenue for the provided service from the consumer of the random number. On the other hand, one part of the above-mentioned revenue will serve as the economic incentive for the producers when the Randaos random number is generated; and the other part will serve as the economic incentive for the Randao community maintenance. The two kinds of incentives ensure Randao will sustainably generate verifiably fair random numbers.

1 The Business Value of the Randao Service

At the current stage, the features of Randaos algorithm can be applied to different situations, including but not limited to quick random number sortition and a random number service for high value lotteries.

1.1 Quick Sortition (random draw)

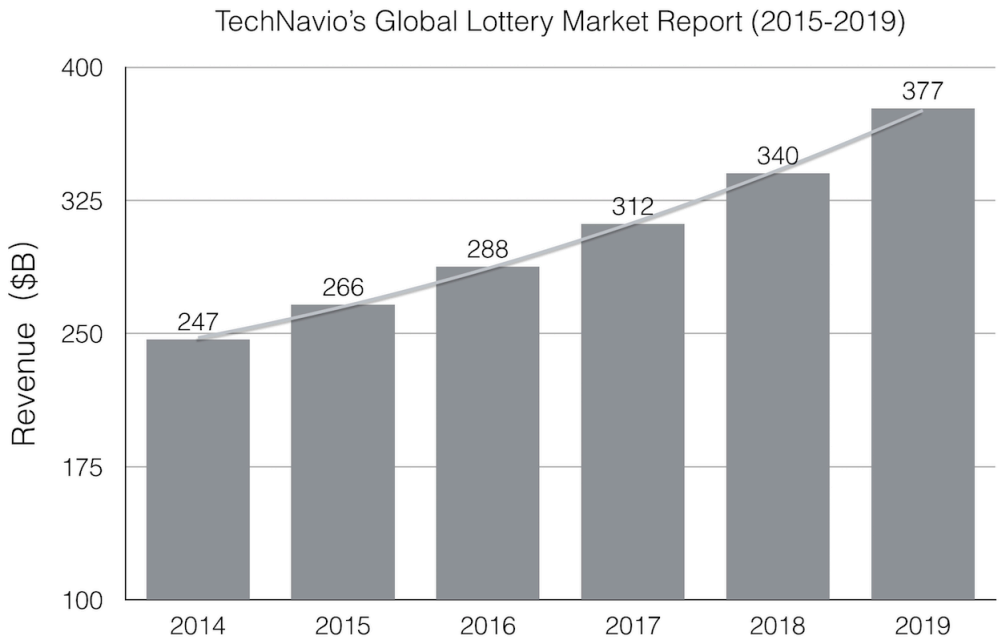
As a relatively fair selection mode, lottery sortition has been widely applied in our daily life. On the basis of traditional sortition, Randao further offers provable fairness, which is of the greatest concerns to participants. The BLS algorithm is very suitable to quickly provide random number generation for high-frequency lotteries with medium or low prizes. Compared with the standard fee[8] of Third-Party Draw Services provided by random.org, it can obtain the minimum revenue of $4.95 (with the cost of $0.0099 per call) and the maximum revenue of over $1149.95 (with the cost of nearly $0.00115 per call) every time it provides the service. Based on the supply and usage frequency of different random number services, the revenue of providing this kind of sortition service can be calculated as follows (Minimum Estimation: sorition every 90s, Medium Estimation: sortition every 60s, Higer Estimation: sortition every 15s):

	Calls	Revenue	Number of Sortition	Annual Revenue
Minimum Estimation	<500	$4.95	350,400	~$1.73M
Medium Estimation	500,000	$649.95	525,600	~$342M
Higher Estimation	1,000,000	$1149.95	2,102,400	~$2.42B

1.2 Super Lotto with High Prizes

Super Lotto and Mark Six are very popular in the market for its high playability, great entertainment and super high first prize. According to the report of TechNavio (TechNavios Global Lottery Market Report 2015-2019), the global lotto market will earn more than $300 billion this year.

[8] https://www.random.org/draws/pricing/



According to the statistics of Business Insider[9], the total market value of all cryptocurrencies reached nearly $100 billion, while M2 reached $83.6 trillion. If the proportion also conforms to the lotto market, it can be inferred that the cryptocurrency in the lotto market amounts to nearly $373 million according to the abovementioned TechNavio report (The lotto market may earn nearly $312 billion in 2017.). However, according to another report[10] from Deloitte Consulting, at least 10% of the global GDP will be saved on the blockchain platform by 2025. If the proportion also conforms to the lotto market, it can be inferred that nearly $59.8 billion will be saved on the blockchain platform by 2025 in view of the nearly 8% annual increase rate in the lotto market. And the majority of them will be formed through the transaction of cryptocurrency.

However, the traditional lotto draw has all along failed to achieve the provable fairness. In order to gain the trust of lottery buyers, they have to adopt methods like notarization at the spot and on live television. Despite that, the fairness of lotto draws has been under suspicion for a long time. The Commit Reveal algorithm of Randao provides a solution for this kind of lottery with low-frequency high prizes. The size of the market can be estimated in the following two ways.

1.2.1 Year-on-year Estimation

According to The WLA Global Lottery Data Compendium 2015[11] of the World Lottery Association (WLA), its members (most legitimate lotto organizations under supervision around the world) earned nearly $159.9 billion from lottos based on raffles in 2014. Among them, $16 billion are from lottos based on raffles of the China Sports Lottery. The sports lotteries that sell all over China include Pai Lie 3, Pai Lie 5, Super Lotto, Selecting 5 from 22, Seven-star Lottery and football lottery. The sports lotteries draw the winning numbers every Monday, Wednesday and Saturday with on-site notarization service provided by the relevant notaries of an escrow company. According to the public information[12], the notarial fee of sports lottery is nearly $2.17 million (RMB 14.75 million yuan). If the notarization service can be replaced by the verifiably fair random number service of

[9] http://www.businessinsider.com/bitcoin-compared-to-all-of-the-worlds-money-2017-6

[10] https://www2.deloitte.com/lu/en/pages/technology/articles/impacts-blockchain-fund-distribution.html

[11] https://world-lotteries.org/media-news/publications/wla-compendia/2077-the-wla-global-lottery-data-compendium-2015

[12] http://www.ccgp.gov.cn/cggg/zygg/zbgg/201609/t20160908_7289856.htm

Randao, it can be estimated that the size of the market that provides the random number service for legitimate lotto organizations under supervision amounts to nearly $21.7 million in proportion.

1.2.2 Frequency Estimation

According to the abovementioned public information of Chinese sports lotteries, it can be calculated that the revenue of every notarization service was about $384. If the verifiably fair random number service offered by Randao can be adopted by the larger-scale lottery with high prizes, assuming that every WLA member organization uses a random number generation service every day, it can be calculated that the corresponding annual revenue is as follows:

	Revenue per Service	Number of Service	Annual Revenue
Estimation	$384	~54,750~150*1*365	~$21M

2 Economic Incentives for Randao Producers

Generally, if the random number generated every phase is called r times on average and costs p ETH each time, the revenue of every phase is rp ETH. Suppose that the random number can be generated with the cooperation of n producers, the cost is $n * 3 * 500 * gasPrice + Ccost$ ($Ccost$ is the gas value consumed internally in the Smart Contract C of Randao, including calculation and storage). Hence, the revenue of every producer is $(rp - 1500 * n * gasPrice - Ccost)/n$ in every service. At present, $gasPrice$ is 25 gwei and C is estimated to consume $1500n$ gas. So the per capita net revenue is estimated to be $rp/n - 7.5 * 10^{-5}$ ETH every service.

2.1 Analysis of Service Pricing Based on Target Revenue

The frequency of random number generation is very high. For example in the business value part of this White Paper, the minimum generation frequency (every 90s) of each quick sortition can achieve the annual linear revenue rate of $0.000001 * 40 * 24 * 365 = 35.04\%$ even if the par rate is only 0.0001% every time.

Suppose that 10 producers participate in every phase of the random number generation with the deposit of 1000ETH, the minimum revenue cannot be lower than 0.01075ETH according to the revenue estimation formula $rp/n - 7.5 * 10^{-5}$ if the revenue rate is higher than 0.0001%. In this case, if the random number is called only 100 times, it costs nearly 0.0001075ETH every time. However, if the random number is called 1000 times, it costs only 0.000001075ETH every time.

2.2 Analysis of Revenue Rate Based on the Number of Calls

At the current real market price (in September, 2017) of $ethPrice = $~$300, it assumes that 20 producers participate in the random number generation every time with the deposit of 10000 ETH for every address and the parameter of 5 per hour. If its price is fixed according to the price list of the Third-Party Draw Service of random.org, it can be analyzed that the revenue rate of the random number generation provider is as follows:

Calls	Revenue	Cost	Profit	Return	Annual ReturnLinear
<=500	$4.95	0.45	4.5	0.000008%	0.3285%
1,000	$8.95	0.45	8.5	0.000014%	0.6205%
2,000	$16.95	0.45	16.5	0.000028%	1.2045%
3,000	$22.95	0.45	22.5	0.000038%	1.6425%
5,000	$34.95	0.45	34.5	0.000058%	2.5185%
10,000	$54.95	0.45	54.5	0.000091%	3.9785%
25,000	$99.95	0.45	99.5	0.000166%	7.2635%
50,000	$174.95	0.45	174.5	0.000291%	12.7385%
80,000	$219.95	0.45	219.5	0.000366%	16.0235%
100,000	$249.95	0.45	249.5	0.000416%	18.2135%
250,000	$399.95	0.45	399.5	0.000666%	29.1635%
500,000	$649.95	0.45	649.5	0.001083%	47.4135%
750,000	$899.95	0.45	899.5	0.001499%	65.6635%
1,000,000	$1149.95	0.45	1149.5	0.001916%	83.9135%

3 Constrains on Randao Economics

Randao serves as infrastructure in the blockchain for other contracts to call. Different contracts have different requirements for random numbers, some require a high level of security, like the results of Super Lotto; some require stability and timeliness, like ordinary contracts where there is no direct interest in the result generated; others require a call-back, which can automatically send a notification after a certain random sequence is generated.

It is obvious that a single contract cannot meet the demand under different scenarios, so several DAO Contracts with different initial parameters will be created, while the basic regulations will remain the same. For a demand of high level of security, the deposit at the first step will be increased significantly, so that the cost to make a generation process fail by not submitting s will increase radically. While for a random number generation contract with lower requirements, the requirements for minimum participants and deposit can be lower.

Lets use an app for betting on parity to illustrate how to reach the high level of security by adjusting different parameters, that is the cost to break the rules will be higher than what you gain. Assuming someone puts 1000 ETH to participate in the parity bet, and calls the random number generation contract C_1, if the generation fails in C_1, wait for the next output, until there is a valid output. Now lets create the contract C_1. There is a deposit requirement of 2000 ETH. If the participant in the bet (marked as G) has also participated in C_1, when he finds an unfavorable situation to himself, he opts to not submit s, so that the generation process fail. Now he loses his 2000 ETH deposit in C_1, and only got 1000 ETH from wining the bet. It is obviously not smart. But G can reduce his lose in C_1 by some means, say, he registered two accounts to participate in C_1 at the same time, and sent two $sha3(s)$, when he meets an unfavorable situation, only one account will refuse to submit s. If there is only one other participant in C_1, G only loses 1000 ETH. He is still expected to win the game, and get 1000 ETH reward. Thus its worth a try.

One solution is to fine the deposit and the fined deposit will not return to participant as rewards. In this way, a contract with 1000 ETH deposit can meet the security requirement in the bet on parity. In addition to this method, another solution eradicates this kind of attacks by introducing an extra scheme Randao Membership. People need to pay the membership fee to hold a membership, and anyone who pays the fee can have his membership. There are different classifications on memberships depending on the fee. The membership doesnt belong to a certain contract, but it is qualified to participate in some of the proof of identity in some contracts. If the member break one of the contracts, membership fee will be fined. Now we can add another line to the contract C_1, that the contract only accepts random numbers submitted by members with a certain membership (say who had paid 1000 ETH or above membership fee), so that no one have the incentive to attack the system.

Part VII

Randao Roadmap

1 Q4 2017

- Launching Commit Reveal random number generation smart contract
- Launching the App on Web

2 Q1 2018

- Launching BLS random number generation smart contract
- Launching the bot service

3 Q2 2018

- Launching the DApp

4 Q3 2018

- Launching the Dev Community