

# DecentRandom: P2P 네트워크 기반 오픈소스 난수발생기

김세환(misskiwi)

(Dated: September 22, 2019)

난수(*亂數, random number*: 규칙성이 없는 무작위 수)는 다양한 분야에 활용되고 있다. 예측 불가능한 난수는, 누군가의 판단을 대신하기도 하고, 공정한 중재의 역할을 하기도 한다. 동전 또는 주사위 던지기부터 경품이나 복권 추첨 등 크고 작은 분야에 난수가 사용된다. 특히 대부분의 서비스가 전산화되면서 온라인 상에서 난수를 활용하는 빈도도 잦아지고 있다. 하지만, 온라인 참여자 모두가 만족할 수 있는 안전한 난수를 생성하여 활용하는 방안은 충분히 진전되지 못하고 있다.

온라인 서비스에 난수를 도입하기 어려운 이유는 첫째, 전산 상에서 난수를 만드는 것 자체가 어렵다는데 있다. 컴퓨터가 쉽게 난수를 만든다고 생각하기 쉽지만, 컴퓨터는 정해진 계산식을 통해 결과를 도출하는데 특화되었기에 무의식적인 행동이나, 우연에 따른 선택을 실행할 수 없다. 컴퓨터는 실행시점의 시간을 천분의 일, 만분의 일초로 나누어 인간이 인지할 수 없을 정도로 잘게 조개 의외성을 가장하여 이를 난수의 용도로 사용한다.

두 번째 문제는, 난수를 요하는 대부분의 서비스가 공정성과 투명성의 보장을 중요시하지만, 정작 온라인 상태에서 이를 담보하기 어렵다는 점이다. 대부분 신뢰받는 제 3자(*trusted third party*)가 권한을 갖고 난수의 생성 과정을 관리하고 감독하는데, 이처럼 특정 집단에 권한이 집중되면 부정이 발생할 소지가 높아진다. 중개자의 부정이 발생하지 않더라도, 간접비용이 증가하는 등 부수적인 문제가 발생하기 쉽다.

이러한 문제를 해결하기 위해 구조적, 확률적으로 안전한 난수를 P2P 네트워크와 블록체인 기반 분산 원장을 통해 효율적으로 생성하고, 모든 과정을 투명하게 유지하는 시스템을 제시하고자 한다.

## I. 서론(INTRODUCTION)

블록체인을 이용하여, 신뢰할 수 있는 제 3자를 두지 않고도 투명하고 안전하게 네트워크를 유지할 수 있음은 비트코인(Bitcoin)[1]의 사례로 충분히 증명되었다. 이는 기존의 중앙화(中央化, centralized)된 비효율적인 구조를 블록체인 기반의 탈중앙화(脫中央化, Decentralized)된 구조로 개선하여 효율성을 제고(提高)할 수 있음을 의미한다.

비트코인을 필두로, 블록체인을 활용한 다양한 시도가 이어지고 있다. 이중 이더리움(Ethereum)[2]은 스마트 컨트랙트(smart contract)<sup>1</sup> 기반 생태계를 구성하여 탈중앙화 응용프로그램(DApp, decentralized application)을 구현할 수 있도록 하였다. 이더리움의 대중화로 탈중앙화 프로젝트들이 급격히 증가하고 있다.

탈중앙화 기조는 양적인 면에서 증가세에 있지만, 질적으로는 아직 갈 길이 멀다. 대부분의 현행 프로젝트들은 블록체인 기반지식이 없는 사용자나, 기존 서비스 환경에 익숙한 이들이 쉽게 사용하기 어려운 구조

로 설계되어 있다. 사용자에게 충분한 편의를 제공하지 못하면 아무리 효율적인 서비스라 하더라도 상용화에 안착하기 어렵다는 것을 간과한 프로젝트가 많다.

우리는 신뢰가 필요 없는(*trustless*) 시스템을 모든 사용자가 쉽게 사용할 수 있도록 제공하는 것을 목표로 한다. 전체 시스템의 논리적 무결성(無缺性, integrity)을 누구나 확인할 수 있도록 공개된 분산 네트워크에 공표(公表)하지만, 최종 사용자(end-user)들은 굳이 이를 이해하고, 검증할 필요가 없다.<sup>2</sup> 탈중앙화의 혜택을 위해 사용자가 많은 것을 감내하게 하면 서비스의 상용화 속도가 느려질 뿐만 아니라, 확산 범위도 제한적일 수 밖에 없다는 점은 이미 여러 탈중앙화 프로젝트에서 찾아볼 수 있다.

효율적이고 안전한 난수를 활용하는 탈중앙화 서비스를 누구나 쉽고 편리하게 사용할 수 있도록 하는 것이 DecentRandom의 최종 목표이다.

<sup>1</sup> 프로그램 코드(code)에 의해 이해가 강제되는 계약

<sup>2</sup> 극단적으로, 최종 사용자들이 블록체인 기반 분산 서비스를 이용하는지조차 모르고 응용프로그램을 사용하기를 바란다.

## II. 이론적 배경 및 선행연구(EXISTING WORKS)

### A. 비트코인(Bitcoin)

2008년, Satoshi Nakamoto에 의해 제안된 비트코인은 신뢰할 수 있는 제 3자가 없더라도 가치를 저장하고 전송할 수 있음을 증명하였다.

기존의 전통적인 온라인송금 방식은 전자서명 알고리즘(DSA, *digital signature algorithm*)을 통해 원격으로 소유권을 확인하고 검증하는 구조였다. 하지만, 이중지불(double spending)<sup>3</sup>을 막기 위해 신뢰할 수 있는 제 3자의 중개(仲介)를 필요로 하기 때문에, 비용이 증가하여 효율성이 떨어지거나 권력의 집중으로 투명성이 결여되는 등 문제가 발생할 소지가 많다. 이에 반하여 비트코인은 암호학(*cryptography*)을 통해 중개자를 거치지 않는 당사자 간 직접거래를 실현하였다.

비트코인의 전자화폐 시스템은 Figure.1과 같이, 디지털서명<sup>4</sup>이 연결된 것으로, 소유자들은 이전 거래와 다음 소유자의 공개키(public key)를 암호화(hash)하여 전자 서명하고, 이들을 연결하여 전송하는 형태로 거래를 구성한다. 수취인은 암호학적으로 소유권을 확인하는 과정을 통해 서명의 진위를 검증할 수 있다.

수취인의 입장에서 원 소유주의 모든 거래가 공개 네트워크를 통해 확인 가능한 상태이면, 중개자가 보증하지 않더라도 이중지불이 발생할 가능성을 회피할 수 있다. 이는 비트코인이 탈중앙화를 유지하는 핵심 요소가 된다.

또한, 비트코인의 거래는 Figure.2과 같이 시간 순으로 블록체인에 기록되고 다음 회차의 블록이 이전 블록의 안정성을 강화하는 형태로 연결되는데, 이 과정이 반복되어 체인이 길어질수록 한 블록의 안정성은 점증적으로 높아지는 특성을 지닌다.

이를 P2P 기반으로 구현하기 위해 비트코인은 Adam Back이 제시한 해시캐시(Hashcash)[3]와 유사한 작업증명(作業證明, *proof-of-work*)을 이용한다. 비

트코인의 작업증명은 해시함수를 실행한 결과가 만족하는 0 비트(bit)를 가질 때까지 임의의 값인 논스(nonce)를 변화하여 반복 계산하도록 강제한다. 반복된 작업의 결과가 주어진 조건을 충족하여 블록이 생성되면, 이 블록은 이제까지 블록 생성에 소요된 작업의 양보다 더 높은 강도의 일을 해내지 않는 한 변경할 수 없다. 네트워크를 유지하는 노드(node)들이 유효한 연결로 받아들이지 않기 때문이다.

이 구조는 보상(incentive) 시스템을 통해 제 3자의 개입이나 중재가 없는 상태에서도 유지된다. 작업자가 작업증명을 통해 블록을 생성하면 네트워크 유지에 대한 보상으로 새롭게 생성된 비트코인과 블록에 포함된 각 거래의 수수료를 합하여 블록을 생성한 작업자에게 제공한다. 유인책이 되는 보상 코인으로 인해 네트워크를 공격하여<sup>5</sup> 얻는 수익보다 네트워크에 협력하여 얻는 수익이 더 크도록 유지한다. 이는 작업자들이 블록체인 네트워크에 순기능을 하도록 하는 동기부여기제(動機附與機制)로 작용한다.

### B. 코스모스(Cosmos)

P2P 기반의 블록체인은 대부분의 경우, 이종(異種) 블록체인 간 폐쇄성이 존재하는 경우가 많다. 체인 간 폐쇄성은 블록체인 응용프로그램의 확산을 저해하는 요소로 작용할 수 있다.

폐쇄성의 문제를 해결하기 위해 아토믹 스왑(atomic swap)이나 사이드체인(side-chain)과 같은 다양한 응용기술들이 시험되어 왔다. 아토믹 스왑은 다중 서명(多重署名, multi signature) 스마트 컨트랙트를 통해 중개자 없이 이종 체인 간 암호화화폐 교환을 가능케 하였으며, 사이드체인은 본(本) 체인의 무결성이 보장되는 상태에서 새로운 체인과의 동기화(同期化)를 구현하여 체인을 유지한다. 하지만 언급한 기술들은 폐쇄성 해소를 위한 지엽적(枝葉的)인 해결책으로, 체인 간 연결의 보편적인 수단으로 활용하기는 무리가 있다.

코스모스(Cosmos)[4]는 이러한 난점을 해결하기 위해, 블록체인 간 연결을 쉽게 처리할 수 있도록 제안된 오픈소스 프로젝트이다.

<sup>3</sup>비트코인 시스템에서는 UTXO(unspent transaction output)를 기준으로 잔고를 체크하는데, 동일한 UTXO를 다른 곳에 중복하여 사용하는 것이 이중지불에 해당한다.

<sup>4</sup>비트코인은 ECDSA(elliptic curve digital signature algorithm)을 사용한다.

<sup>5</sup>공격은 잘못된 블록을 생성하는 행위를 의미한다.

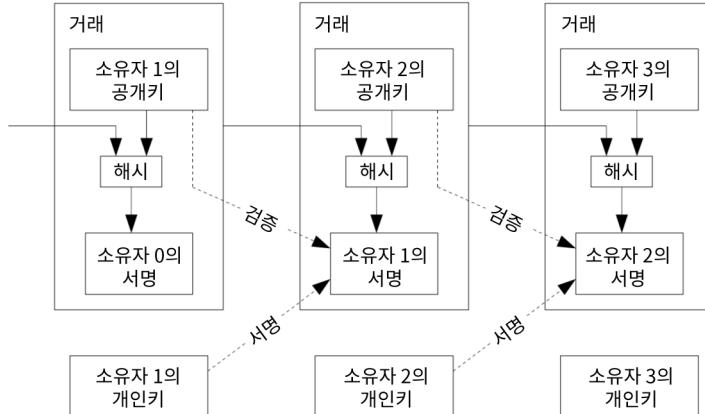


Figure 1. 비트코인 거래의 구성<sup>a</sup>

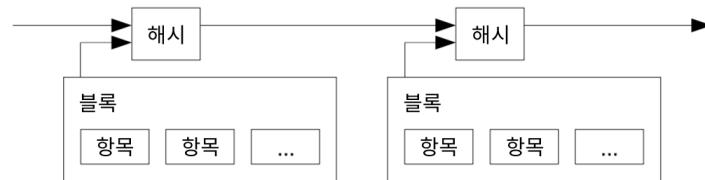


Figure 2. 비트코인 블록의 연결 방식<sup>a</sup>

코스모스는 블록체인 간 연결을 구현하기 위해 표준화된 SDK<sup>6</sup>의 *ABCI(application blockchain interface)*를 통해 블록체인을 제어한다. 이를 통해 표준화된 각각의 체인을 존(zone)이라 하며, 이들은 IBC(inter blockchain communication) 프로토콜을 사용하여 허브(hub)나 다른 존과 상호연결할 수 있다.

체인 간 연결을 용이하게 처리한다는 장점 외에도, 코스모스 SDK를 사용하면 텐더민트라는 검증된 합의 알고리즘 기반의 고성능 블록체인을 수월하게 만들 수 있다.<sup>7</sup> 텐더민트는 비잔틴장애허용(Byzantine fault tolerance)을 적용한 지분증명(持分證明, proof-of-stake) 합의 알고리즘(consensus algorithm)을 통해 블록체인을 구성하는데, 다른 합의 알고리즘을 사용하는 블록체인에 비해 상대적으로 빠르게 데이터를

처리할 수 있다. 또한, 대부분의 블록체인이 확률적 완결성을 보장하는 블록을 생성하는데 반해 텐더민트는 완결성을 지닌 확정(確定) 블록을 생성한다는 장점을 지닌다. 이는, 여타 블록체인 시스템보다 안정적인 데이터 처리가 가능함을 의미한다.

체인 간 연결을 쉽게 처리하더라도,  $n : n$  연결을 체인이 추가될 때마다 구성하는 것은 비효율적인 방법이다. 필요한 작업량은  $n$ 이 늘어날 때 지수(指數)의 형태로 증가하기 때문이다. 코스모스는 허브를 통해 이와 같은 비효율성을 제거한다. 각각의 존을 IBC 프로토콜을 통해 허브와 연결하여 모든 체인을 일대일 연결하는 수고를 들이지 않고도 존과 허브, 존과 존 간에 상호 통신을 가능케 한다.<sup>8</sup>

코스모스는 활용하면 DecentRandom을 상호 연결이 가능한 고성능 개방형 블록체인으로 만들 수 있다.

<sup>6</sup> 이를 코스모스 SDK라 한다.

<sup>7</sup> 2019년 5월 기준, 코스모스 SDK는 텐더민트 알고리즘을 기반으로 한 ABCI만을 제공하고 있다.

<sup>8</sup> 네트워크 연결을 관리하는 하드웨어 허브의 역할과 같다.

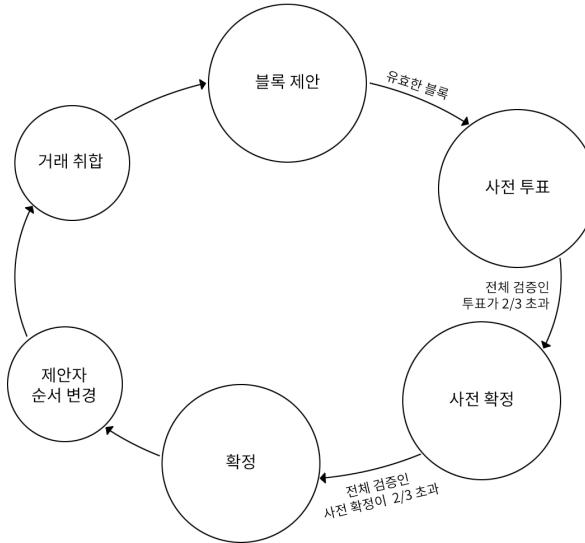


Figure 3. 텐더민트 코어의 블록생성과정

### C. 순수 난수(True Random Number)

순수 난수는 규칙성을 찾을 수 없는 예측 불가능한 난수를 뜻한다. 순수 난수를 판단하는 기준은 상대적인 경우가 많다. 가령, 자연에서 발생하는 잡음(noise)은 현재의 기술 수준에서 순수 난수로 활용할 수 있으나, 기술의 발달로 잡음 생성과정의 전후 패턴을 완전히 규명할 수 있게되면, 이는 더 이상 순수 난수라 할 수 없다.

순수 난수는 시드(seed)나 논스(nonce) 같은 변수 없이 그 자체로 안정성을 담보할 수 있어야 한다.

순수 난수는 이상적인 난수이나, 계측과 처리가 불가하기 때문에 이를 서비스에 활용하기는 어렵다.

### D. 의사 난수(Pseudo Random Number)

의사 난수(擬似亂數)는 완전히 예측 불가능한 난수라 할 수 없다. 의사 난수를 만드는 데 사용한 함수와 투입 변수를 모두 알 수 있다면, 오차 없이 예측이 가능하다.

하지만, 이 예측을 불가능에 가깝게 처리하면, 확률적으로 허용되는 범위 내에서 안전한 값을 얻을 수 있고, 이는 난수의 역할을 대신하여 사용하기 충분할 수 있다. 특히, 컴퓨터는 우연에 근거한 예측 불가능한

함수를 구현하기 어렵기 때문에 의사 난수를 빈번하게 사용한다.

의사 난수에 대한 연구는 광범위하게 이루어져 왔다.

#### 1. 선형합동생성기(線形合同生成機, Linear Congruential Generator)

선형합동생성기는 재귀(再歸) 관계로 정의된 수열을 순차적으로 활용하여 수를 선택하고, 이를 통해 얻은 결과를 난수로 사용한다.

선형합동방식을 사용하는 가장 대표적인 사례는 C 언어의 stdlib.h에 포함된 srand(), rand() 함수이다. C 언어의 ISO9899에 포함된 의사 난수 생성에 사용되는 코드는 다음과 같다.

```

1 // ISO9899
2 static unsigned long int next = 1;
3 int rand(void)
4 {
5     next = next * 1103515245 + 12345;
6     return (unsigned int)(next/65535) %
32768;
7 }

```

C언어의 rand()는 난수를 선형적(線形的)으로 선택하는 기능을 하는 함수이고, srand()는 수열을 선택하는 기능을 한다. srand()를 통해 수열을 선택하고

rand()를 통해 선형선택을 하는 것이 일반적인 사용법이다.

```
1 #include <time.h>
2 #include <stdlib.h>
3
4 srand((unsigned int)time(NULL));
5 int number = rand();
```

선형합동방식은 사용이 간편하고, 자원소모가 적어 효율적인 난수 생성이 가능하다. 하지만, 선택의 근거가 되는 수열의 변화에 한계가 있고, 사용 가능한 자료형(資料型)의 범위가 한정되어 있다는 단점도 존재한다. 소규모 작업에 필요한 난수를 생성하여 사용하기 충분하지만, 규모가 크거나 중요도가 높은 분야에 활용하기에는 무리가 있다.

## 2. 메르센 트위스터(*Mersenne Twister*)

메르센 트위스터[5]는 Makoto Matsumoto와 Takuji Nishimura에 의해 제안된 알고리즘으로, C언어를 기반으로 제작되었다.

메르센 트위스터는 반복된 주기의 난수를 발생하는 의사 난수 생성기이다. 메르센 소수(素數)를 사용한다는 점에서 일반적인 의사 난수 생성기와 차이가 있다. 메르센 트위스터를 통해 생성된 난수는 623차원까지 동일분포(同一分布)되어 있고, 비트(bit) 연산을 통해 구현이 가능하여 속도가 빠르다. 선형합동법으로 난수를 생성하면 64비트 기기에서 가능한 최대주기가  $2^{64}$ 에 불과하지만, 메르센 트위스터는  $2^{19937}-1$ 에 달하는 긴 주기의 난수를 사용할 수 있다. 이 외에도 메르센 트위스터는 난수의 선택기준이 되는 시드(seed)를 파일로 저장하여 사용할 수 있는데, 이는 원격상황에서의 검증에 유용하게 활용할 수 있다.

생성기가 필요로 하는 메모리(memory)는 선형합동 생성기에 비해 큰 편이나, 극단적으로 소형화된 임베디드(embedded)환경이 아니라면 크게 문제될 정도는 아니다.

여러 장점에도 불구하고, 메르센 트위스터 또한 의사 난수의 일종으로 완전한 난수라고 할 수는 없다. 난수의 특성<sup>9</sup>을 알고 있으면 수열을 도출할 수 있고, 이를

통해 다음 난수를 예측하는 것이 가능하기 때문이다.

의사 난수를 그대로 탈중앙화 프로젝트에 적용하기는 무리가 있을지라도, 난수의 선택을 위해 수열을 생성하고, 시드를 활용하는 방식은 온라인 환경에 적용하기 좋은 구조라 할 수 있다.

## E. 개선된 난수 생성 알고리즘

앞서 순수난수를 판단하는 기준은 상태성을 띠고 있다고 밝힌 바 있다. 개선된 난수생성알고리즘들은 상대적으로 순수난수에 가깝고, 사용하는 조건에 따라 순수난수로 볼 수도 있지만 대부분이 암호학, 수학적인 안전장치를 필요로 한다는 점에서 순수난수와 대비된다.

### 1. 하드웨어 난수 발생기

하드웨어 난수 발생기는 예측이 불가능한 정보를 하드웨어를 통해 수집한 뒤, 이를 소프트웨어 처리과정을 거쳐 난수의 형태로 바꾸는 도구이다. 예를 들면, 주변 환경의 잡음을 센서를 통해 수집하고, 전산화된 데이터를 변형하여 난수를 추출하거나, 불규칙적으로 흔들리는 불꽃을 찍은 화상 데이터를 난수로 바꾸는 방법 등이 이에 해당한다.

### 2. NIST 비콘(Beacon)

NIST 비콘(national institute of standards and technology beacon)은 양자역학(量子力學)을 이용하여 난수를 생성한다. 때문에 NIST 비콘을 통해 생성된 난수는 양자역학적으로 안정성이 보장된다.

NIST비콘은 알버트 아인슈타인(Albert Einstein)이 언급한 원격에서 발생하는 으스스한 활동(spooky action at a distance)을 자료화하여 난수생성에 사용한다. 으스스한 데이터(spooky data)와 5천만 번이 넘는 벨 테스트(bell test)를 루프홀 없이(loophole-free) 진행한다. NIST 비콘은 측정장치나 중간과정에 사용된 시드가 알려지더라도, 결과 값은 여전히 난수라는 것을 증명할 수 있다.

<sup>9</sup>난수생성에 사용된 주기와 범위

하지만 NIST 비콘이 생성한 난수도 벨 테스트가 물리적(物理的) 안정성을 보장 받는다는 전제에 한하여 안전하다.

#### F. 탈중앙화 난수 생성에 대한 기존연구

순수 난수와 의사 난수 모두 다자간(多者間) 온라인 공유 시스템에 활용하기 어렵다. 원격 네트워크의 특성상, 개체간 시간과 공간이 불일치하는 경우가 많은데, 순수 난수는 물론 의사 난수도 이러한 상황에서 모든 네트워크 참여자에게 안정성과 투명성을 보장하는 난수를 제시하기 어렵다. 이러한 이유로 원격 네트워크 상에서의 난수를 생성하는 과정에는 신뢰받는 제 3자가 중재 및 감독의 역할을 담당하는 경우가 많다.

중앙화된 중재자의 존재는 비효율성과 지나친 권한의 편중이라는 문제를 야기하기 마련이다. 이러한 문제를 탈중앙화된 방식으로 해결하고자 하는 연구가 꾸준히 진행되어왔다.

##### 1. 탈중앙화 의사 난수 생성 알고리즘

Serguei Popov는 신뢰를 기반으로 하지 않고, 중재자의 개입이 없는 방식으로 다수의 그룹이 만족할 수 있는 탈중앙화 난수 생성기를 제시하였다.[6]

각각의 그룹은 높은 불확실성을 가지는 난수를 생성하고, 이를 다른 참여자가 알아볼 수 없도록 암호화하여 네트워크에 공개한다. 그리고 각 그룹 내에서 다시 소 그룹을 묶고 이들 간에 난수를 공유하도록 한다. 이 과정을 성공적으로 수행한 그룹이 원래 선택한 숫자를 공개하고 이들 간 배타적논리합(排他的論理合, XOR) 연산을 수행하여 난수를 생성한다.

비밀을 공유하는 과정에서 협조하지 않는<sup>10</sup> 참여자는 최종 연산에서 제외된다.

이 알고리즘은 중개자 없이 원격의 참여자들이 납득할 수 있는 난수를 만들어냈다는 점에서 의미가 크다. 하지만 많은 수의 참여자가 존재할수록 네트워크를 공격할 수 있는 여지가 줄어드는 구조 임에도, 참여자의 온라인 상태를 강요하는 불편한 방식이 결과적으로

<sup>10</sup>참여자가 오프라인(off-line)인 상태여도 이 경우에 해당한다.

참여를 저해하고 네트워크의 안정성에 해를 끼칠 수 있다는 점은 개선할 필요가 있다.

##### 2. RanDAO

RanDAO[7]는 난수를 생성하는 탈중앙화된 자율 조직(DAO, decentralized autonomous organization)으로 이더리움 시스템 위에 스마트 컨트랙트로 배포되어 작동한다.

난수 생성에 참여하는 이들은  $m$  만큼의 ETH<sup>11</sup>와 숫자  $s$ 를 sha3 함수로 암호화하고, 결과값 sha3( $s$ )를 일정기간<sup>12</sup>동안 정해진 계약<sup>13</sup>에 저장한다. 정해진 기간이 지나면, 참여자들은 암호화 과정을 거치지 않은  $s$ 를 스마트컨트랙트에 인증한다.<sup>14</sup> EVM(Ethereum virtual machine, 이더리움가상머신)상에서 처음 보낸 해시와 나중에 보낸  $s$ 를 sha3()로 해시한 값이 일치하는지 검사하여 일치하는 값은 저장하고 일치하지 않는 값은 저장하지 않는다. 그리고 마지막으로 검증이 끝난  $s$  값들의 배타적논리합을 계산하여 난수를 생성하는 방식이다.

이 방식 또한 Serguei의 연구와 마찬가지로, 구성원의 지속적인 참여를 필요로 한다는 점에서 실용화가 어렵다는 문제가 있다.<sup>15</sup>

### III. 구성(DESIGN)

기존 연구에 대한 고찰의 결과로, 우리는 아래의 결론에 도달할 수 있다.

- 순수 난수를 온라인 서비스에 적용하는 것은 불가능하다.
- 수학적, 암호학적으로 안전한 난수에 대한 연구는 충분히 다각도로 진행되었으며 온라인 서비스에 충분히 활용할 수 있다.

<sup>11</sup>Ether, 이더리움에서 사용되는 화폐단위

<sup>12</sup>기간은 블록을 기준으로 한다.

<sup>13</sup>이더리움의 스마트컨트랙트

<sup>14</sup>이 방식은 기본적으로 Serguei Popov의 알고리즘(목차.II F 1)과 유사하다.

<sup>15</sup>이는 결국 이더리움의 네트워크에서 수수료로 사용되는 가스(gas)를 과도하게 소모하는 문제와 직결된다.

- 원격 네트워크 상에서 난수 생성을 중재자에 의존하게 되면 비효율성, 불투명성 등의 문제가 발생한다.
- 중개자를 배제하고 탈중앙화 구조를 유지하기 위해서는 참여자에 대한 동기부여가 필요하다.
- 많은 이들의 참여를 독려하기 위해, 사용자 환경은 구조적으로 간단하고 명료해야 한다.

DecentRandom은 기존 연구들을 토대로 상용화가 가능한 탈중앙화 난수 시스템을 제시하고자 한다. 근간이 되는 블록체인 시스템은 수학적, 암호학적으로 안전함을 증명할 수 있어야 하고, 참여하는 이들이 전문적인 기반지식이 없어도 쉽게 시스템을 이용할 수 있어야 한다. 네트워크의 안정성은 오픈소스의 형태로 공개된 코드와 이에 대한 증명, 검증을 통해 확보한다.

DecentRandom의 구조는 블록체인 시스템과 이를 쉽게 이용할 수 있도록 돋는 공개 *SDK*(*software development kit*)로 구분할 수 있다. 블록체인 시스템은 *RAND* 토큰(이하 *RAND*)을 통해 유지되는 P2P 네트워크이며, 신뢰받는 제 3자를 필요로 하지 않는다. *RAND*를 통해 위임자와 난수요청자, 블록을 생성하는 검증인(*檢證人, validator*) 모두가 각자의 유인책에 따라 스스로 동기부여되어 네트워크를 유지한다.

### A. 블록체인(Blockchain)

DecentRandom의 블록체인은 코스모스 SDK(목차.II B)를 기반으로 제작한다.

#### 1. 합의 알고리즘(*Consensus Algorithm*)

DecentRandom은 코스모스 SDK에서 제공하는 ABCI를 통해 텐더민트(Tendermint)기반 블록체인을 운용한다. DecentRandom 존은 IBC 프로토콜을 통해 코스모스 허브 내의 다른 존과 상호연결이 가능하다.<sup>16</sup>

DecentRandom 블록체인은 비잔틴장애허용 지분증명을 사용하는 텐더민트의 특성을 그대로 따른다.

---

<sup>16</sup> 2019년 5월 현재, 코스모스 SDK는 IBC 프로토콜은 개발이 진행 중이며, 이후 정식으로 코스모스 SDK에 포함될 예정이다.

검증인들은 자신의 지분과 위임된 지분의 합에 비례하여 블록을 생성할 권한을 갖는다. 블록 생성과정은 텐더민트 코어(core)에 의해 안전하게 처리된다.

검증인은 순서에 따라 블록을 제안(*proposal*)한다. 블록이 제안되면 검증인단은 제안된 블록에 사전투표(*pre-vote*), 사전확정(*pre-commit*)을 진행하여, 블록을 확정(*commit*)한다. 이 때 안정적인 합의는 전체 노드의  $\frac{2}{3}$ 의 찬성을 기준으로 한다. (Figure.3 참조)

DecentRandom 블록체인은 *RAND* 토큰(이하 *RAND*로 표기)을 기반으로 유지된다. 검증인은 자신의 *RAND*와 위임자가 위탁한 *RAND*를 담보로 블록을 확정하고 이에 대한 보상을 *RAND*로 지급 받는다. 위임자는 검증인에게 자신의 *RAND*를 위탁하여 네트워크에 참여하고 보상을 받는다. 마지막으로, 난수요청자는 안전한 난수를 공급받는 대가로 네트워크에 수수료를 지불한다.

DecentRandom 내에서 블록체인은 크게 두 가지 기능을 한다. 하나는 DecentRandom의 통화(通貨)인 *RAND*의 소유권 이동과 저장의 역할이며, 다른 하나는 난수 발생과 관련된 상태(state)를 저장하는 상태머신(state machine)의 기능이다.

### 2. 모듈(Modules)

코스모스 SDK는 블록체인 제작을 위해 기본 모듈을 제공한다. 아래 열거된 모듈은 DecentRandom에서 사용하는 Cosmos 모듈이다.

- **genaccounts:** 제네시스 계정 모듈
- **genutil:** 제네시스 관련 기능을 위한 모듈
- **auth:** 계정과 수수료 관련 기능을 위한 모듈
- **bank:** *RAND*의 관리를 위한 모듈
- **crisis:** 위험 관리를 위한 모듈
- **distr:** 분배를 위한 모듈
- **gov:** 거버넌스를 위한 모듈
- **params:** 파라메터 관리를 위한 모듈
- **mint:** 토큰 생성을 위한 모듈

- slashing: RAND 삭감을 위한 모듈
- staking: RAND의 위탁과 관련된 모듈
- supply: 토큰 공급 관리를 위한 모듈

우리는 이 외에 DecentRandom을 위한 특화된 모듈을 제작할 필요가 있다.

- rand: DecentRandom에 난수 관련 데이터를 활용하기 위한 모듈<sup>17</sup>

우리는 이를 통해 키밸류(Key-value) 저장 방식<sup>18</sup>의 상태머신(*state machine*) 형태로 DecentRandom을 구현한다.

### 3. 상태(*State*)와 메시지(*Messages*)

RAND의 상태는 auth와 bank 모듈을 통해 관리한다. 이 외에 난수 라운드(round)의 상태는 rand 모듈에서 관리한다.

상태변화를 야기하는 메시지는 거래(transaction)에 포함된다. DecentRandom에서 사용하는 주요 메시지는 다음과 같다.

- MsgNewRound: 난수 라운드를 생성하는 메시지
- MsgDeployNonce: 논스 검증을 위한 메시지
- MsgAddTargets: 모집단 입력을 위한 메시지
- MsgUpdateTargets: 모집단 데이터 수정을 위한 메시지

거래가 블록에 포함되어 노드에 전달되면, ABCI를 통해 메시지를 해석한다. 그리고 해석한 결과를 상태머신에 기록한다.

---

<sup>17</sup><https://github.com/decentrandom/decentrandom/x/rand>  
<sup>18</sup><https://godoc.org/github.com/cosmos/cosmos-sdk/types#KVStore> 참조

### 4. 거버넌스(*Governance*)

DecentRandom의 거버넌스방식은, 코스모스 SDK의 형식을 따른다. 검증인은 운영 변수(operational parameters)의 변경을 위한 제안에 투표를 통해 의사를 표시할 수 있으며 검증인은 모든 제안에 투표해야한다.

또한, 네트워크 상으로 확인이 불가능한 검증인의 부정(不正)이 발견될 경우, 거버넌스를 통해 해당 검증인의 권한을 박탈할 수 있다.

### 5. RAND

RAND는 DecentRandom 내에서 P2P 분산원장으로 관리되는 전송 가능한 디지털화폐이다. RAND는 디지털서명을 통해 소유권을 증명할 수 있으며, 이는 소유권 이전의 유효성을 판단하는 근거가 된다.

DecentRandom 내에서 RAND가 하는 역할은 크게 두 가지이다.

#### 1. 전송을 위한 소유권 이전 및 확인

#### 2. 전송과 데이터 기록에 소요되는 수수료

RAND는 일반적인 암호화화폐(暗號化貨幣, crypto-currency)와 마찬가지로 제 3자의 중개 없이 전송이 가능하다. 코스모스 허브의 DecentRandom 존에 존재하는 RAND는 여타 폐쇄된 블록체인과는 다르게 IBC 프로토콜을 통해 코스모스 허브 내의 다른 암호화화폐와 쉽게 교환이 가능하다는 장점을 가진다.<sup>19</sup> RAND는 코스모스 허브를 이용함으로써 기존의 암호화화폐 보다 나은 체인 간 가치교환 환경을 제공한다.<sup>20</sup>

가치의 전송 이외에도 RAND의 전송이나 난수 관련 데이터의 기록 등 블록체인 상에서 이뤄지는 일련의 작업을 제 3자의 개입 없이 처리하기 위한 수수료의 기능으로도 사용한다. 블록체인에 데이터를 기록하고 이에 대한 유효성을 보증하는 행위를 정확하고 안전하게 처리하려면 검증인에게 제공할 유인책이 필요하다.

---

<sup>19</sup>물론, 별도의 중앙화된 거래소를 이용하더라도 가치교환은 가능하지만, 이 경우 거래소의 해킹, 부정행위 등의 문제나 입, 출금 비용이 지나치게 높게 책정되는 문제에서 자유롭지 못하다.

<sup>20</sup>2019년 9월 기준, IBC 프로토콜은 개발 중에 있다.

검증인이 난수 생성에 기여할 경우 보상으로 RAND를 제공하고, 난수 생성에 해를 가하여 자기 소임을 다하지 못할 경우 반대급부로 이들의 RAND를 차감하는 조건을 통해 순기능하도록 하여, 안정적인 난수 시스템을 유지한다.

## B. 참여자

현재 전산환경에서 주로 사용되는 의사난수 생성방식은 순수난수를 사용할 수 없는 온라인 환경에 적용할 수 있는 효율적인 대안이다. 하지만, 의사 난수는 아래의 문제로 인해, P2P 네트워크에 적용하기 어렵다.

- 난수 생성기의 사용 권한을 누가, 그리고 누구에게 부여할 것인가?
- 블록체인의 특성상 발생할 수 있는 시간차나 전파 속도의 차이가 난수의 안정성을 해치지 않는가?

특정 집단에게 권한이 집중되면 이로 인해 문제가 발생할 수 있음을 앞서 기존 연구에서 확인한 바 있다. 또한 원격 네트워크 상에서 블록 단위로 확정되는 분산 원장(分散元帳)은 블록이 확정되고 배포되기 까지 시간을 필요로 하기 때문에, 전파 속도가 다른 참여자들 간 정보의 불균형을 야기할 수 있는 점 또한 감안해야 한다.

이런 이유로 의사난수의 장점을 취하되, 이를 P2P 네트워크에 사용 가능하도록 새로운 난수 생성 방식을 구현할 필요가 있다.

아래 전제조건을 통해 논의를 확장하도록 하자.

- 난수를 요청하는 난수요청자와 난수생성에 필요한 근거를 제공하고 난수 관련 정보를 블록에 확정하는 검증인이 있다.
- 난수 생성과정에 다수가 참여한다.
- 결과가 도출되기 전에는 참여자 및 외부의 누구도 이를 예측할 수 없어야 한다.
- 결과가 도출된 이후, 네트워크 구성원은 물론, 외부의 불특정 다수가 생성과정의 공정성을 검증할 수 있어야 한다.
- 생성된 결과는 변복할 수 없어야 한다.

### 1. 검증인(Validator)

검증인은 블록을 제안하고 확정하는 역할을 한다. 이들은 자신이 보유한 RAND를 예치하고, 위임자에게서 RAND를 위임받아<sup>21</sup> DecentRandom 네트워크를 안정적으로 유지한다. 이에 대한 대가로 자신이 생성한 블록마다 RAND를 보상으로 받는다. 반대로, 잘 못된 블록을 생성하거나<sup>22</sup> 일정 시간 동안 정상적으로 블록을 생성하지 못하면, 예치한 RAND가 삭감(削減, slash)되도록 강제하여 정상적인 블록을 생성하도록 독려(督勵)한다.<sup>23</sup> DecentRandom 생태계 내에서 블록의 확정은 난수 생성 단계의 확정을 포함하기 때문에 거래 확정 이상의 중요성을 가진다.

검증인은 블록을 제안할 때, 난수생성에 사용할 시드(seed)<sup>24</sup>를 정한 뒤, 이를 암호화하여 네트워크에 알린다. 그리고, 다음에 자신이 블록을 제안하는 회차에, 직전에 암호화하여 공표한 시드의 암호화하지 않은 원래 값과, 새로 생성한 시드의 해시 값을 블록 헤더에 포함한다.

이 과정에서 검증인은 암호화하지 않은 시드의 값을 블록에 공개하기 전까지 어느 누구에게도 누설해서는 안된다. 만약 검증인이 이를 어겨 시드를 알게 된 누군가가 이를 블록체인 네트워크에 인증하면 검증인이 예치한 RAND를 삭감하여 시드를 인증한 이에게 지급한다. 이는 검증인이 시드의 비밀을 유지하도록 강제하는 조건으로 작용한다.

Figure.4에서 검증인 1은 블록 높이  $n$ 에서 시드  $S_{1_1}$ 과  $S_{1_2}$ 의 암호화한 값을 알린 뒤, 다음 자기 회차인  $n + 1$  높이에서 암호화하지 않은 시드  $S_{1_1}$ 을 네트워크에 공개해야 한다.

만약, 공개한 시드를 sha3() 함수로 암호화한 값이 이전 블록에 담긴 암호화한 시드와 일치하지 않는다면, 해당 검증인의 RAND를 삭감한다.

<sup>21</sup>코스모스 SDK에서는 이를 본딩(bonding)이라 한다.

<sup>22</sup>동일한 높이에 두 개의 블록에 서명하거나 유효하지 않은 블록을 제안한 경우

<sup>23</sup>이에 대한 자세한 내용은 단락.IV A 1 b에서 살펴보도록 한다.

<sup>24</sup>임의의 값을 사용한다.

## 2. 위임자(Delegator)

위임자는 검증인을 지정하여 RAND를 위탁하여 자격을 얻게 되며, 위탁을 철회하는 순간 자격을 상실한다.

위임자는 검증인에 RAND를 위탁하고, 검증인의 부정을 견제하며, 이에 대한 대가로 검증인과 수수료를 공유한다. 이들은 언제든 위탁을 행하거나 철회할 수 있다.<sup>25</sup>

위임자에게 지급되는 보상은 거래 수수료와 DecentRandom의 인플레이션 생성량에서 검증인 수수료를 제하고 위탁한 비율에 따라 지급된다.

검증인이 네트워크에 악영향을 끼치면 검증인은 물론 해당 검증인에게 위탁한 위임자의 RAND도 함께 차감되기 때문에, 위임자는 검증인 선택에 신중을 기해야 하며, 주기적인 감시를 행해야 한다. 검증인이 부정한 행위를 하거나 이를 감지한 경우 위임자는 위임 철회 행동을 통해 검증인에 대응할 수 있다.

## 3. 난수요청자(Random Number Requester)

난수요청자는 DecentRandom 네트워크에 난수, 혹은 난수를 이용하는 처리 결과를 요청하고, 결과를 제공받는 대가로 RAND를 지불한다. 난수 요청은 블록 단위로 처리된다.

검증인이 임의의 시드를 설정하여 난수 연산의 기초를 제공하듯, 난수 요청자는 검증인이 결과를 예측할 수 없도록 논스(nonce)를 선택해야 한다. 논스는 검증인이 만든 시드와 함께 결과를 도출하는 기준이 되는 값이기 때문에, 라운드가 종료되기 전까지 검증인 측에서 이를 알 수 없어야 한다. 비밀을 유지하기 위해, 난수요청자는 요청 시점에 논스를 그대로 공개하지 않고 sha3()로 암호화한 해시만 블록체인에 알린다. 검증인이 시드를 인증한 뒤, 요청자가 결과를 개봉할 때 해시되지 않은 원래의 논스 값을 인증하여 결과를 도출한다. 이 때, 요청자가 전파한 암호화된 논스와 비교하여 일치하지 않으면 해당 라운드를 무효(invalid)로 판명한다.

<sup>25</sup>단, 철회 시 21일 동안 RAND의 전송이 제한되는 위임철회기간(unbonding period)이 있다.

Figure.4에서 난수요청자의 요청 R1은 검증인 2가 제공한 K3를 난수요청자가 지정한 논스 N1을 대입하여 결과를 얻는다.

난수요청자가 논스를 먼저 선택하고 나서 검증인의 시드 생성이 이루어지므로, 논스의 변화가 최종 결과에 어떻게 영향을 줄 지 예측하지 못하게 하는 효과가 있다. 검증인의 입장에서 결과를 선택하는 데 사용되는 논스를 모르는 상태에서 불완전해를 도출하는 것이기 때문에, 자신의 선택한 시드가 어떤 결과를 가져올지 미리 알 수 없다.

난수요청자는 자신이 요청하는 난수의 안정성을 담보하기 위해 난수발생보증금을 예탁할 수 있다. 난수 발생보증금은 난수 요청시 추가할 수 있으며 보증금은 해당 라운드가 정상 종료되는 시점에 요청자의 계좌로 반환되지만, 무효처리 되거나 다른 이가 자신의 논스를 네트워크에 인증하면 돌려받을 수 없다.

## C. 난수 생성 단계

DecentRandom이 제안하는 기본적인 난수 생성 단계는 Figure.4와 같다.

이를 구체화하면 아래와 같다.

### 1. 요청(Request)

난수요청자는 요청을 거래(transaction)의 형태로 네트워크에 전파한다. 그리고 요청은 검증인에 의해 블록에 포함되어 확정된다.

난수요청자가 취할 수 있는 요청의 종류는 다음과 같다.

1. 라운드 개설

2. 모집단(母集團) 등록

3. 결과 도출(라운드 해결)

라운드(Round)는 난수의 요청시점부터 결과를 도출하기까지의 전체 과정을 뜻하는 용어이다. 라운드는 난수요청자에 의해 시작되고 검증인의 시드 생성을 거친 후, 요청자의 논스 서명을 거쳐 완료된다. 각 라운드는 고유한 번호를 부여 받는데 이를 라운드 아이디(round ID)라 한다. 이는 라운드의 구별에 사용한다.

	검증인 1	검증인 2	요청자
Block $n$	$S1_1$ $S1_2$		
Block $n+1$	$S2_1$ $S2_2$ $S1_1$		
Block $n+2$		$S3_1$ $S3_2$	
Block $n+3$	$S4_1$ $S4_2$ $S1_2$ $S2_1$		
Block $n+4$	$S5_1$ $S5_2$ $S2_2$ $S4_1$		
Block $n+5$		$S6_1$ $S6_2$ $S3_1$	
Block $n+6$	$S7_1$ $S7_2$ $S4_2$ $S5_1$		$N1$
Block $n+7$	$S8_1$ $S8_2$ $S5_2$ $S7_1$		$S7_1$
Block $n+8$		$S9_1$ $S9_2$ $S3_2$ $S6_1$	$S3_2$
			$N1$

Figure 4. 단순화한 라운드 전개 과정

기존 탈중앙화 난수 발생기(목차.II F)는 별도의 요청자가 없는 불특정 다수를 위한 난수를 발생하는데 반하여, DecentRandom은 난수요청자의 요청에 의해 각각의 개별 라운드가 실행된다. 이는 수요와 공급의 원칙에 따라, 수요자가 지불한 비용이 공급자에게 대가로 전해지는 명료한 경제 구조를 만들고, 불필요한 네트워크 자원소모를 줄이는 효과가 있다.

라운드 개설 시 요청자는 라운드의 성격을 규정해야 한다.

### 1. 난수 난이도

### 2. 라운드 시작 시점

### 3. 난수발생보증금 규모

난수 난이도(*難易度, difficulty*)는 1 이상의 정수(*整數*)를 사용한다.<sup>26</sup> 난이도는 몇 회 이상의 생성 단계를 거치는지<sup>27</sup>를 규정하는데, 난이도가 높을 수록 여러 차례 검증인을 거치게 된다. 고찰.2에서 언급하겠지만, 난이도는 난수의 안정성을 나타내는 척도가 되며, 클 수록 안전하다.<sup>28</sup>

<sup>26</sup>난이도 1의 난수를 요청할 수 있으나, 이는 특정 상황에서는 안정성면에서 권장되지 않는다. 3 참조.

<sup>27</sup>이는 몇 개의 블록을 거치는지와 같다.

<sup>28</sup>난이도는 안정성을 나타내는 상대적인 척도이므로, 절대지표로 활

요청자는 라운드의 실행 시점을 지정하여 예약 실행 할 수 있다. 기본값은 자연이 없는 즉시 진행 방식이나, 미래의 특정 시점을 지정하면 해당 시점 이후 가장 먼저 생성된 블록을 기준으로 한다.

또한, 난수발생보증금을 추가하여 안정성을 제고할 수 있다. 난수발생보증금은 난수요청자가 자신의 논스를 안전하게 유지할 것임을 증명하는 일종의 담보물로, 금액이 높을수록 안전하다. 난수발생보증금을 설정하는 것은 요청자의 선택사항이다.

난수요청자는 위에 언급한 설정 내용과 sha3()로 암호화한 논스를 네트워크에 전파한다. 난수요청자는 최종 결과를 받아보기 위해 논스를 분실하지 않아야 하며, 이를 절대 다른 이와 공유해서는 안된다. 논스의 비밀을 유지하는 책임은 난수요청자에게 있다. 요청자가 최종 논스를 인증하기 이전에 누군가 먼저 논스를 네트워크에 인증하면 해당 라운드가 무효화되는 동시에 난수발생보증금을 논스를 인증한 이에게 지급한다.

모집단을 필요로 하는 라운드가 개설된 경우, 요청자는 모집단을 블록에 기록하는 요청을 할 수 있다. DecentRandom은 투명성과 공정성을 부여하기 위해, 모집단을 블록체인 네트워크에 기록할 수 있다. 만약 모집단을 필요로 하지 않는 경우 이는 생략할 수 있다.

마지막으로 요청자는 결과를 얻기 위해, 라운드 개설 시 선정한 논스를 밝히는 요청을 네트워크에 보내는 것으로 라운드를 종료한다.

위 과정은 시간차를 두고<sup>29</sup> 진행된다.

## 2. 검증인의 시드(Seed) 생성

시드는 라운드의 개설 여부와 상관없이 매 블록 검증인이 수행해야하는 의무이다. 검증인은 매 블록 새로운 임의의 값을 시드로 생성하고 이를 sha3()로 암호화하여 블록 헤더에 포함해야 한다. 이 때 시드는 현재 네트워크의 검증인 수 만큼 생성해야한다.

Figure.4는 아래 조건에서의 시드 생성 방식을 단순화한 것이다.

용할 수 없다. 같은 수치의 난이도라도 네트워크의 상태(검증인의 수나 검증인 보팅 파워(voting Power) 분포 등)에 따라 척도가 달라지기 때문이다.

<sup>29</sup>블록체인 상에서 시간차라 함은 각기 다른 블록 높이(height)에 요청하는 것을 뜻한다.

- 총 2명(검증인1, 검증인2)의 검증인이 존재한다.
- 검증인1의 보팅 파워(블록 생성 능력)는 검증인2의 두 배이다.

검증인은  $S1_1, S1_2$  두개의 시드를 생성하고 블록헤더에 sha3( $S1_1$ ), sha3( $S1_2$ )를 포함한다. 다음 블록 생성 순서에는 새로  $S2_1, S2_2$  두개의 시드를 만들고 블록헤더에는  $S1_1, sha3(S2_1), sha3(S2_2)$ 를 포함한다. 그 다음 회차에는 다시  $S3_1, S3_2$  두개의 시드를 만들고, 네트워크에  $S1_2, S2_1, sha3(S3_1), sha3(S3_2)$ 를 추가하여 블록을 생성한다.

여러개의 시드를 생성하는 이유는 난이도 개념을 도입하기 위함이다. 난이도가 적용된 라운드의 진행 방식은 뒤에서 좀 더 살펴보도록 하겠다.

## 3. 최종 시드 도출

난이도의 회차 만큼 시드 생성이 진행되면, 난수 생성의 기준이 되는 최종 시드를 도출할 수 있다. 이 단계는 아직 논스를 적용하지 않은 상태로 최종 결과를 알 수 없으므로 우리는 이를 불완전해(不完全解)라 부른다.

난수 생성에 불완전해를 사용함으로써 DecentRandom은 기존의 탈중앙화된 난수 생성기(목차.IIF)와 차별화된다. 기존 연구는 난수 생성에 참여하는 이들이 자신이 선택한 난수를 네트워크에 공개하고 이를 증명하는 과정을 지속적으로 반복해야 한다. 이는 참여자의 입장에서 비효율적이고 불편한 방식이며, 난수를 필요로 하는 다양한 분야에 적절히 활용하기 어렵다. DecentRandom은 최종 결과의 도출이 아니라 불완전해를 만드는 작업까지만 검증인이 담당하도록 하여 위임자는 지속적인 행동을 취하지 않아도 난수 생성 과정에 참여할 수 있다. 이는 참여율을 높여, 네트워크를 더 안전하게 유지하는데 도움을 준다.

## 4. 종료(Finalize)

라운드가 종료 되면 난수요청인이 라운드 개설 시에 선정한 논스와 검증인의 시드를 조합하여 최종 결과를 얻는다. 이 과정에서, 요청자가 정상적인 논스를 인증하였는지에 대한 검증은 네트워크에 전파할 수도,

그렇지 않을 수도 있다. 블록체인 외부에 논스를 공개 하더라도 정상적인 논스를 인증했는지 확인할 수 있기 때문이다. 다만 난수발생보증금을 예탁한 경우 논스 인증절차를 거쳐야만 보증금이 환원된다.

라운드가 종료되는 형태는 다음과 같다.

- **파기:** 요청자의 논스가 유출된 경우
- **불완전 종료:** 정상적으로 불완전해가 도출되었으나, 난수 요청자가 논스를 증명하는 행동을 정해진 기간동안 취하지 않아 불완전하게 종료되는 경우
- **정상 종료:** 정상적으로 불완전해가 도출되고, 난수 요청자가 정상적인 논스로 인증한 경우
- **비정상 종료:** 불완전해 도출이 실패하거나, 난수 요청자가 잘못된 논스로 인증한 경우

정상 종료 및 불완전 종료인 경우 다음의 단계를 거쳐 최종 결과를 산정한다.<sup>30</sup>

1. 불완전해 도출에 성공한 검증인들의 시드 값을 취합한다.
2. 난수요청자의 논스와 검증인의 시드를 파서에 대입하여 결과를 얻는다.

블록체인에 기록된 데이터를 기반으로 난수를 도출하는 것은 오픈소스로 공개된 파서를 통해 처리하므로 네트워크의 자원을 소모하지 않는다.<sup>31</sup> 때문에, 최종 결과를 위해 요청자의 논스를 네트워크에 공개하지 않더라도, 정확한 논스만 알면 누구나 블록체인 외부에서 난수를 도출할 수 있다.

난수 요청자가 논스를 인증하지 않아 불완전 종료되는 것이 꼭 문제라고 볼 수는 없다. 다만, 이 경우 난수요청자는 결과 값과 함께 자신이 선택한 논스를 P2P 네트워크가 아닌 다른 방식으로 모든 이에게 공개하여 확인 가능하도록 해야한다. 이를 공개하지 않으면 요청자는 논스를 무작위로 대입하여 결과를 바꾸는 행동을 할 수 있기 때문이다.

<sup>30</sup>이 작업은 블록체인 네트워크 상의 합의가 아닌 파서(parser)를 통해 이루어진다.

<sup>31</sup>이는, 최종 결과를 도출하기 위해 P2P 네트워크에 별도의 요청을 보낼 필요가 없음을 뜻한다.

DecentRandom은 불완전 종료를 권장하지 않지만, 상대적으로 위험도가 낮고, 빠른 처리를 요하는 난수의 생성 등에 유용하게 활용할 수 있으므로, 이에 대한 선택은 사용자의 판단에 맡기도록 한다.

Figure.4를 통해 난이도를 2로 설정한 정상 종료된 라운드가 어떻게 난수를 추출하는지 확인할 수 있다.

난수요청자가  $n + 6$  블록에 라운드를 개설하며 논스  $N_1$ 을 설정하고 이를 암호화하여 공개하면, 다음 블록 높이인  $n + 7$ 에서 검증인1이 공개한 시드  $S_{7_1}$ 을 얻을 수 있다. 여기까지의 과정이 난이도 1에 해당한다. 난이도 2를 처리하기 위해 다음 블록인  $n + 8$ 가 생성되면 검증인2가 공개한 시드  $S_{3_2}$ 를 얻을 수 있다. 그리고 난수요청자가 자신의 논스  $N_1$ 을 공개하면 두 개의 시드와 하나의 논스를 파서에 대입하여 난수를 추출한다.

## 5. 활용 및 검증

라운드가 정상 혹은 불완전 종료되면, 블록체인에 기록된 정보를 기반으로 난수를 도출하여 활용할 수 있다. 블록체인에 기록된 정보는 인간이 해독할 수 없는 해시의 형태이므로, 이를 인식 가능한 난수로 분석(Parse)하는 작업이 필요하다. 이 때 사용하는 분석기를 파서(parser)라 한다.

파서의 처리 구조에 따라 최종 결과가 달라질 수 있으므로, 원칙적으로 DecentRandom에서는 공식적으로 단일 파서만을 지원한다. 다만 필요에 따라 독자적으로 제작한 커스텀 파서를 사용할 수도 있으나, 이 경우 파서의 코드 전체를 라운드 시작 전에 확인 가능하도록 공개해야 한다.

## IV. 유인책과 보안(INCENTIVES AND SECURITY)

### A. 유인책

#### 1. 검증인의 동기부여

검증인은 블록을 생성하기도 하고, 난수의 선정기준이 되는 불완전해를 도출하는 역할도 한다. 이는 DecentRandom 네트워크를 안전하게 유지하기 위한 중

요한 역할이다. 따라서, 검증인이 부정한 행동을 하지 않도록 하는 것이 중요하다. 탈중앙화된 DecentRandom 네트워크는 보상과 벌칙을 통해 검증인이 바른 행동을하도록 유도하는데, 이는 검증인이 경제적으로 합리적이라는 가정하에 성립한다.

a. 보상(*Incentives*) DecentRandom에서 검증인이 보상으로 RAND를 받는 경우는 아래와 같다.

- 거래 수수료

- 블록 보상

거래 수수료는 거래(*transaction*)를 전파하는 이가 네트워크에 지불하는 비용이다. 검증인은 미확정거래 저장소(*mempool*)에 모인 거래들을 모아 유효성(有效性)을 검사하고 블록에 포함하는데, 거래 수수료는 검증인이 해당 거래를 우선적으로 블록에 포함하도록 유인(誘引)하는 역할을 한다.

블록 보상은 검증인의 동기부여와 RAND 인플레이션 조절에 이용된다. 검증인과 위임자가 네트워크에 위탁하는 RAND가 증가하면 인해 시장 유통량이 감소하는 성향을 보인다. 이를 보완하기 위해 전체 네트워크 위탁량에 따라 보상을 조절하여 적절한 유동성을 유지한다. 또한, 블록 보상은 검증인의 입장에서 가장 기본적인 수익원으로 블록 생성의 동기가 된다.

상기 언급한 보상유형은 위임자와 나누게 되며, 이에 대한 수수료율은 검증인이 책정할 수 있다.

검증인은 자신의 수익을 극대화하기 위해 가능한 많은 위임자가 자신에게 RAND를 위탁하길 원한다. 보상의 분배 비율을 높게 유지하는 것은 더 많은 위탁을 받기 위한 좋은 유인책이지만, 이에 못지않게 네트워크를 안정적으로 운영하는 능력을 증명하는 것 또한 위임자를 유치하는 중요한 매개이다. 검증인이 잘못된 블록을 생성하거나 난수 생성에 부정적 영향을 끼칠 경우 검증인의 RAND는 물론, 위임자가 위탁한 RAND도 삭감되기 때문에 위임자는 검증인 선택 시에 높은 수익률과 동시에 안정성을 함께 고려하게 된다.<sup>32</sup>

---

<sup>32</sup> 이와 별도로, 위임자의 입장에서 총 위탁량이 높은 비중을 보유한 검증인을 선택하는 것도 적절치 못하다. 다량의 RAND를 위탁한 검증인에 위임하더라도, 평균 보상액은 큰 차이가 없으며, 오히려 공격자나 해커가 많은 RAND를 보유한 검증인을 공격 대상으로 삼아 위험도가 증가하기 때문이다.

b. 벌칙(*Penalties*) 검증인의 역할이 중요한 만큼, 이들이 잘못된 행동으로 네트워크에 해를 끼쳤을 때 단호한 조치를 취해야 한다. 여기서 조치라 함은 위탁한 RAND를 삭감(*slash*)하는 행위이다.

DecentRandom에서 검증인이 벌칙으로 자신의 RAND를 삭감당하는 경우는 다음과 같다.

- 잘못된 블록을 생성할 경우
- 일정 시간동안 정상적인 블록을 생산하지 못할 경우
- 정해진 시간에 처리해야 할 난수 라운드를 처리하지 못한 경우
- 암호화되지 않은 시드 값이 블록에 인증되기 전에 유출될 경우
- 검증인 거버넌스에 의해 삭감이 필요하다고 판단한 경우

검증인은 동일한 블록 높이에 두 개 이상의 블록에 서명해서는 안된다. 이는 합의구조를 위반하여 네트워크를 혼란스럽게하는 행위이다. 따라서 우리는 이 경우, 검증인과 그에게 위임한 위임자의 RAND를 삭감하여 문제가 발생하지 않도록 미연에 방지한다.

블록의 생성<sup>33</sup> 또한 검증인의 중요한 임무이다. 검증인이 이를 제대로 수행하지 못하면 네트워크는 지연되고, 최악의 경우 정상적인 기능을 할 수 없다. 이 경우 검증인의 RAND와 해당 검증인에 위탁된 RAND를 삭감하여 검증인이 이러한 행동을 피하도록 독려한다.

검증인이 생성하는 시드 값은 난수 발생의 중요한 변수(變數)이므로, 시드 값을 안전하게 관리하는 것은 매우 중요하다. 만약 누군가 검증인이 다음 차례 블록에 원래의 시드 값  $S$ 를 인증하기 전에 정확한 값을 인증하면, 검증인의 RAND를 삭감하여 인증한 이에게 보상으로 지급한다. 난수 생성 역할은 그만큼 중요도가 높기 때문에 검증인이 이를 지키도록 하기 위해 강도 높은 벌칙을 부여하는 것이다.

마지막으로, 검증인이 네트워크의 상태 변화로는 확인할 수 없는 잘못<sup>34</sup>을 저지르고 이 문제가 밝혀지는

---

<sup>33</sup> 특정 검증인이 자신의 차례에 블록을 제안하고 네트워크 합의를 통해 제안이 받아들여지는 일련의 과정

<sup>34</sup> 현실 세계에서의 담합, 뇌물 수수 등

경우, DecentRandom의 거버넌스를 통해 해당 검증인의 RAND를 삭감하거나 심할 경우 검증인 자격을 박탈할 수도 있다.

## 2. 위임자의 동기 부여

위임자의 역할은 블록 생성 과정에 자신의 RAND를 위탁하여 블록의 안정성을 담보하고 검증인의 부정을 견제하는 일이다. 블록 생성에 기여한 부분은 검증인이 제안한 분배비율에 따라 달라진다. 위임자의 RAND 보상 조건은 검증인의 경우(단락.IV A 1 a)와 같고, 위탁한 RAND가 삭감되는 조건 역시 검증인의 기준(단락.IV A 1 b)을 따른다.

위임자는 보상을 극대화하고 삭감을 피하기 위해 검증인을 견제하는 역할을 한다.

## B. 난수 안정성에 영향을 주는 요인

### 1. 난이도

각각의 난수 라운드는 1 이상의 정수로 구성된다. 난이도는 총 몇 개의 시드를 이용할 것인지를 뜻한다. 모든 블록은 해당 시점의 검증인의 수 만큼 시드를 포함하고 있다. 난이도가  $n$ 인 라운드는  $n$ 개의 블록 높이 동안 시드를 추출하게 된다.

난이도를 증가시켜 여러 검증인의 시드를 사용하게 되면 난수의 불확실성(不確實性)이 증가하여 안정성이 높아지는 효과가 있다.

### 2. 적절한 검증인의 수

네트워크의 투명성만 고려한다면 검증인의 수는 많을수록 좋다. 하지만 필요 이상으로 많은 검증인은 오히려 합의 효율성에 영향을 줄 수 있으므로, 검증인의 수를 적정 수준으로 제한할 필요가 있다.<sup>35</sup>

검증인 제한은 이후 거버넌스를 통해 변경할 수 있다.

---

<sup>35</sup>DecentRandom은 초기 단계에서 검증인의 수가 100명이 넘지 않도록 제한하고자 한다. RAND의 위탁량을 기준으로 100번째에 들지 못하면 블록 제안에 참여할 수 없다.

## 고찰 1. 검증인의 구성과 이에 따른 난수 안정성의 변화

검증인 구성에 따라 DecentRandom 네트워크의 안정성은 어떻게 변하는지 살펴보도록 하자. 아래 조건의 변화에 따라 검증인 구성은 달라진다.

- 검증인의 수

- 검증인이 위탁한 RAND의 분포

우선, 전체 검증인의 수  $N$ 은 초기에 100명을 넘지 못한다. 또한 블록 생성을 위해 최소 1인의 검증인을 필요로 하기 때문에,

$$1 \leq N \leq 100 \quad (1)$$

이다.

검증인이 위탁한 RAND의 분포가 고르다고 가정할 경우, 난이도  $d$ 로 인해 각기 다른 검증인을 거치는 유의미한 범위  $r_d$ 는,

$$1 \leq r_d \leq \min(d, N) \quad (2)$$

이다. 난이도  $d$ 를 높이더라도 검증인 수가 난이도보다 작으면 그보다 많은 검증인을 거쳐갈 수 없기 때문이다. 물론, 난이도는 위임자의 수나 RAND의 시장가치 등 다른 변수를 통해 난수의 안정성에 영향을 주지만(고찰.2 참조) 실제 검증인을 거치는 효과  $r_d$ 는 위와 같다. 이를 통해 난수 라운드를 안정적으로 운영하고자 한다면 검증인의 수가 많을수록 좋다는 것을 확인할 수 있다.

검증인에 위탁된 RAND 수령<sup>36</sup>의 분포는 난이도에 어떤 영향을 주는지 살펴보도록 하자.

검증인  $n$ 이 위탁받은 RAND의 양을  $v_n$  일때, 검증인  $n$ 이 블록을 제안할 확률  $p_n$ 은 다음과 같다.

$$p_n = \frac{v_n}{\sum_{n=1}^N v_n} \quad (3)$$

---

<sup>36</sup>이는 검증인의 블록 제안 능력, 즉 보팅 파워(voting power)에 해당한다.

전체 검증인의 평균 위탁량을 기준으로 표준편차를 계산하면 다음과 같다.

$$\sqrt{\frac{(\sum_{n=1}^N \frac{v_n - N}{p_n})^2}{N}} \quad (4)$$

표준편차가 클수록 난이도  $d$  회차 동안 같은 검증인이 블록을 생성할 확률은 높아진다. 이를 최소화하려면 각 검증인의 위탁량  $v_n$ 이 고르게 분포되어야 함을 알 수 있다.

위임자들은 특정 검증인에 위탁량이 집중되는 것이나 수 생성 시스템의 비효율성을 증가시킨다는 점을 인지하고 위탁량이 특정 검증인에 쏠리지 않도록 신중을 기할 필요가 있다. 비단 비효율성의 문제가 아니더라도, 위탁량 독점으로 인해 공격자의 목표가 되는 등의 부작용이 있으므로 지양되어야 한다. 특히, 텐더민트 합의는 전체 보팅 파워의  $\frac{1}{3}$ 이 악의적으로 사용될 경우 문제가 발생할 소지가 있으므로 rand 모듈의 안정성이 전에 텐더민트 ABCI의 안정성을 위해 독점 비율은  $\frac{1}{3}$ 을 넘지 않도록 제한할 필요가 있다.

### C. 담합에 대한 저항

난수의 요청에서 발생까지 난수요청자와 위임자, 검증인들은 각자의 역할을 정상적으로 수행해야 한다. 이 세 참여자가 담합(談合)할 경우 발생 가능한 문제가 없는지 상황별로 살펴보기로 하자.<sup>37</sup>

#### 1. 검증인의 담합

검증인은 시드를 도출하는 역할과 동시에 블록을 생성한다. 때문에, 검증인이 네트워크에 해를 가하지 않도록 설계하는 것이 다른 무엇보다 중요하다.

블록 생성과 관련된 행동을 제외한 난수 발생 관련 검증인의 행위는 다음과 같다.

#### 1. 도출한 시드 값의 해시를 네트워크에 알리는 행위

#### 2. 암호화되지 않은 시드 값을 네트워크에 알리는 행위

검증인이 시드 값의 해시를 블록에 포함하여 제안하지 않으면, 해당 검증인이 보유한 RAND를 차감한다. 시드 값의 해시를 네트워크에 알린 이후, 암호화되지 않은 원래의 시드를 네트워크에 알리지 않을 때에도 RAND를 삭감한다.

위 책임을 이행하지 않았을 때 검증인이 받는 벌칙은 해당 검증인에 위탁된 RAND의 삭감이다.

검증인은 요청자의 논스를 알지 못하므로 시드의 변격에 따라 어떤 각각의 라운드가 어떤 결과를 얻을지 알 수 없다. 검증인의 담합으로 가능한 실질적인 공격은 라운드에 대한 보이콧이다. 텐더민트의 합의 원칙에 따라 검증인이 정해진 순번에 블록을 생성하지 못하는 회수가 늘면 블록 제안 권한을 박탈하고, 이 문제가 지속되면 RAND를 삭감한다. 검증인이 라운드를 보이콧하여 DecentRandom 네트워크 외부에서 얻는 대가가 이보다 클 경우 검증인은 담합할 수 있다고 볼 수 있다.

검증인의 보이콧을 저해하는 요소는 다음과 같다.

- 블록 보상을 놓침으로써 발생하는 기회 비용
- 블록 생성 자연으로 인한 벌칙

검증인은 특정 라운드를 보이콧할 수 없고 블록 전체를 보이콧 할 수밖에 없다. 때문에 검증인은 블록을 생성하여 받는 보상(단락.IV A 1 a) 전체를 기회 비용으로 소진하게 된다.

또한, 블록 제안을 지속적으로 누락할 경우, 벌칙(단락.IV A 1 b)을 받게 되고, 이를 지속하면 검증인 자격을 상실할 수도 있다.

위 두 요소는 직접적으로 보유한 RAND의 손실을 가져오지만, 이 외에도 간접적으로 검증인의 신뢰도를 떨어뜨려, 기존 위임자의 이탈이나 신규 위임자 유입 저하를 가져온다.

텐더민트 코어의 블록 생성 주기는 현재 평균 6.5초 정도이다.<sup>38</sup> 전체 검증인이 담합하여 블록 생성을 10회

<sup>37</sup> 위임자와 난수요청자는 블록 생성 권한이 없기 때문에 둘 만의 담합으로 네트워크에 해를 가하는 것이 불가능하므로, 검증인이 포함되지 않은 담합은 논의에서 제외하였다.

<sup>38</sup> 2019년 2월 26일, Gaia-12001 기준

지연하더라도 1분 5초 정도가 지연되며, 이 보이콧을 계속할 경우, 네트워크 합의에 따라 검증인은 지위를 잃게 된다.

단순히 검증인, 혹은 검증인들 만의 담합으로 난수 혹은 추첨 결과에 치명적인 위해를 가할 수 없으며, 오히려 이로 인해 발생하는 실질적인 피해가 크기 때문에 경제적으로 합리적인 검증인은 담합에 참여할 이유가 없다.

## 2. 난수요청자와 검증인의 담합

앞서 목차.IV C 1에서 살펴본 바와 같이 검증인들 간의 담합만으로는 난수 생성 결과를 조작할 수 없다. 난수요청자와 검증인이 담합하는 경우를 살펴보도록 하자.

검증인과 난수요청자가 담합하면 특정 라운드의 논스와 난수열 생성에 사용하는 시드를 서로 공유하여 알 수 있다.

난이도가  $d$ 인 경우  $d$ 회 블록을 생성하는 검증인들이 상대방의 시드 값과, 요청자의 논스를 모두 공유해야 하기 때문에 난이도가 높아질 수록 담합은 어려워진다. 상대 검증인의 시드 값을 먼저 네트워크에 인증하여 상대 검증인의 위탁 RAND를 가져오는 공격이 가능하므로 검증인 중 누구하나라도 배신하는 경우 담합은 실패한다.

우리는 이를 기준으로 난이도  $d$ 에 따라 담합에 저항하여 보장 가능한 금액을 계산할 수 있다.

### 고찰 2. 난이도에 따른 안정성 보장

난이도가  $d$ 이고 1 RAND의 시장 가치가  $f$  USD이며,  $N$  명의 검증인들이 평균  $s$  RAND를 위탁하고 있는 경우를 가정해보자. 공개된 시드의 원 값을 인증할 경우 해당 시드를 암호화한 검증인으로부터  $x$  비율 만큼의 위탁된 RAND를 취할 수 있다.

위 조건 하에, 검증인 한 명이 담합에 응하여 손해를 입을 수 있는 금액은 다음과 같다.

$$f \cdot s \cdot x \quad (5)$$

검증인은 담합의 대가로 제시되는 금액이 위 금액보다 작다면, 담합에 공모하지 않을 것이다. 이는 역으로

공격자의 입장에서 공격을 통해 얻을 수 있는 금액이 기도 하다. 만약 난수요청자가 검증인을 포섭하고자 한다면 검증인 모두에게 이 금액 이상을 제시해야 한다. 그렇지 않으면 배신을 통해 취하는 금액이 이를 상회하여 담합이 아닌 배신이라는 행동을 취할 수 있고 결국 담합은 실패로 이어지기 때문이다.

난이도가  $d$ 라면, 난수요청자는  $(f \cdot s \cdot x) \times d$ 보다 큰 금액을 각각의 검증인에게 제시해야 한다. 때문에 난수요청자의 입장에서 검증인을 포섭하는데 지불해야하는 최소비용은 다음과 같다.

$$(f \cdot s \cdot x) \times d \quad (6)$$

위 계산의 결과는 DecentRandom 네트워크가 라운드 당 감당할 수 있는 금액이다. 추첨으로 인한 당첨액의 규모가 이보다 작다면(담합을 실행하는데 드는 비용이 당첨액 보다 크다면) 요청자는 검증인과 담합 할 이유가 없기 때문이다.

DecentRandom 네트워크가 감당할 수 있는 라운드 당 처리 금액의 변화는 다음 조건을 따른다.

- RAND의 시장 가치에 정비례 한다.
- 검증인의 시드 노출 시 삼감되는 비율에 정비례 한다.
- 검증인의 수에 정비례 한다.
- 난이도에 정비례 한다.
- 위탁된 RAND의 규모에 정비례 한다.

이를 지수화(指數化)하여 사용하면, 각각의 라운드가 감당할 수 있는 금액을 확인할 수 있다. 이를 라운드 보증한도액(保證限度額)이라 하며, 라운드의 전전성을 가늠하는 데 사용할 수 있다. 가령, 특정 라운드의 라운드 보증 한도액이 50,000 USD이고, 총 당첨액이 100,000 USD라면, 해당 라운드는 보증 한도액이 당첨액 보다 부족하기 때문에 전전하지 못하다고 판단할 수 있다.

라운드 보증한도액은 DecentRandom 네트워크의 활성화에 따라 달라진다. 만약 DecentRandom의 활성화 정도가 부족하여 라운드 보증한도액이 충분하지

않은 상황이라면 난수요청자는 난수발생보증금을 네트워크에 예치하여 보증 총액을 높일 수 있다. 난수발생보증금이 설정된 경우, 요청자의 논스를 제 3자가 네트워크에 인증하면 난수 요청자는 보증금을 돌려받을 수 없게 된다. 난수요청자가 검증인과 담합을 시도하면 검증인의 입장에서 요청자의 난수발생보증금을 가로챌 수 있으므로, 난수요청자는 검증인과 담합에서 불리한 위치를 점하게 되고, 이는 담합을 가능하도록 하는 금액의 상한을 보증금액 만큼 낮추는 역할을 한다.

### 3. 난수요청자인 동시에 검증인이 경우

앞서 다룬 고찰의 증명은 난수요청자와 검증인이 다른 인물이거나 같은 이해관계에 있지 않은 경우에 유효하다. 만약 난수요청자와 검증인이 동일인이거나 같은 이해관계에 속해있어, 정상적인 견제의 기능이 작동하지 않는 경우를 살펴보도록 하자.

난수요청자와 검증인이 동일인이면 난수요청자의 논스를 아는 검증인이 시드 생성에 참여하게 된다.

#### 고찰 3. 난이도가 1인 난수

난이도가 1이고, 논스를 아는 검증인  $v$ 의 보팅 파워가  $p$ 라면, 난수요청자이자 검증인  $v$ 가 요청한 라운드를 자신이 처리하게 될 확률은  $p$ 이다. 하지만 텐더민트 합의 알고리즘의 특성 상, 제안자 우선순위(proposer priority)를 예측하여 다음 차례의 블록제안자가 누구인지 어느 정도 예측이 가능하기 때문에 난이도 1인 난수의 안전은 완벽하게 보장할 수 없다.

이 경우 조작에 소요되는 비용은 기존 네트워크의 가장 낮은 보팅 파워의 검증인이 보유한 RAND 위탁량보다 커야 한다. 최소한 블록을 생성할 수 있는 검증인 지위를 유지해야하기 때문이다. 하지만 현실적으로 보팅 파워는 지속적으로 변동하기 때문에, 정확히 자신이 원하는 시간의 블록 제안권을 획득하기 어렵다.

이런 이유로 난이도 1인 난수는 선택적으로 사안이 중요하거나, 규모가 큰 이벤트에 사용되는 난수로 적합하지 않다.

#### 고찰 4. 난이도가 2 이상인 난수

난이도  $d$ 가 2 이상인 경우,  $d$ 회 만큼의 블록을 독점했을 경우에만 난수를 조작할 수 있다. 이 경우 검증인

$v$ 의 보팅 파워  $p$ 가 어떤 수준인가에 따라 조작의 가능성성이 달라진다.

텐더민트 합의 알고리즘은 제안자 우선순위를 통해 블록 제안자를 선택한다. 보팅 파워가 높을 수록 제안자 우선순위는 빠르게 회복되고, 낮을 수록 복원이 느린다.

조작을 시도하는 검증인이 누구인지 특정할 수 없으므로 전체 네트워크에서 조작이 가능한 난이도의 최대값은 다음과 같다.

$$\left\lceil \frac{\max p_n}{1 - \max p_n} \right\rceil \quad (7)$$

$\max(p_n)$  값은 검증인의 수가 많고, 편차가 고를수록 낮아지는 특성을 지닌다. 하지만 텐더민트 합의 알고리즘의 특성 상 가장 높은 보팅 파워가 33%를 넘지 않도록 유지하고자 할 것이므로, 위 수치는 실제 1을 넘을 수 없다.<sup>39</sup>

다시 말해 네트워크가 정상적인 분포의 검증인을 유지하는 한, 난이도 2 이상의 난수는 검증인인 동시에 난수요청자라 하더라도 조작에 성공할 수 없음을 의미한다. 조작에 성공하기 위해서는 보팅 파워를 확보한 추가적인 검증노드를 운영해야 한다.

#### 고찰 5. 연결된 블록의 검증인이 동일인일 확률

$p_a, p_b$ 의 보팅 파워를 가진 같은 이해관계의 검증인  $v_a$ 와  $v_b$ 가 연결된 2개의 블록을 연달아 생성할 확률은

$$p_a \times p_b \quad (8)$$

이다.  $p_a$ 와  $p_b$ 의 합은 최대치는  $\frac{1}{3}$ 이므로, 최적의  $p_a$ 와  $p_b$ 의 배분은 균등하게  $\frac{1}{6}$ 씩 배분하는 것이다. 이를 일반화하면 같은 이해관계의 검증노드  $n$ 이  $p_n$ 의 보팅 파워로 연달아  $N$  개의 블록을 생성할 확률은 다음과 같다.

---

<sup>39</sup>텐더민트의 합의는  $\frac{2}{3}$ 의 동의를 기반으로 하기 때문에  $\frac{1}{3}$  이상의 악의적 행동은 텐더민트 합의 알고리즘의 안정성에 결함을 가져오게 된다.

$$\prod_{i=2}^N p_i \quad (9)$$

이를 토대로 공격자가 취할 수 있는 최적의 배분은  $N$  개의 검증노드에 각각  $\frac{1}{3}$  의 보팅파워를 할당하는 것이다. 따라서 위 확률은 다음과 같이 단순화할 수 있다.

$$\left( \frac{1}{3 \cdot n} \right)^n \quad (10)$$

이를 통해 난이도의 증가는 같은 블록을 공격자가 독식할 확률을 단순히 선형적으로 감소시키는 것이 아니라, 기하급수적으로 감소시킬 수 있다. 이를 통해 높은 난이도의 난수를 조작하기 위해서는 검증인 노드 대부분을 장악하지 않고서는 불가능함을 확인할 수 있다.

이를 고찰. 2에서 도출한 라운드 보증한도액과 함께 안전한 난수 추출의 보장 근거로 사용할 수 있다.

## V. 결론(CONCLUSION)

난수 혹은 난수를 활용한 서비스는 다양한 분야에서 쉽게 찾아볼 수 있다. 난수를 활용하는 이유는 다양하지만, 대부분 공정하고 투명한 선택을 취하기 위

함이다. 이제까지는 이를 신뢰를 기반으로 제 3자의 통제 하에 처리해왔다. 문제는 신뢰와 권한을 독점한 제 3자에게서 발생한다. 공정성과 투명성의 문제로 난수 선정 과정에 대한 불신이나 모집단의 관리에 대한 불신이 생겨나기도 하고, 비효율성이 높아져 시스템 전체의 비용이 증가하기도 한다.

비트코인에서 시작된 블록체인 기반 탈중앙화 시스템은 중앙화된 시스템의 문제점을 해결할 수 있으며 이는 난수 활용 시스템에도 그대로 적용할 수 있다. 신뢰받는 제 3자를 배제하고 구성원들이 역할에 따라 순기능을 함으로써 네트워크를 유지하는 P2P 분산 시스템을 만들면, 난수 활용 시스템의 공정성과 투명성, 효율성을 모두 높일 수 있다.

본 문서에서 간결하고 명료한 구조의 P2P 난수 활용 시스템을 구현할 수 있음을 증명하고자 하였다. 기존의 난수와 관련된 연구를 기반으로 난수 고유의 특성은 유지하고 탈중앙화의 특성을 더하여 DecentRandom을 구성하였다. 이렇게 구성된 DecentRandom은 원격 네트워크 상에서 보다 안전하고 효율적인 난수 사용 환경을 만들고, 이를 이용하는 서비스 제작자에서 최종 사용자까지 모든 참여자의 효용을 제고할 수 있다.

## ACKNOWLEDGMENTS

끝으로, 이 프로젝트가 나올 수 있도록 기틀을 닦아 준 선구자들과 백서 작성에 도움을 준 모든 이들께 감사의 인사를 전합니다.

- 
- [1] S. Nakamoto. Bitcoin: a peer-to-peer electronic cash system. <https://bitcoin.org/bitcoin.pdf>, 2008.
  - [2] V. Buterin. A next generation smart contract and decentralized application platform. <https://github.com/ethereum/wiki/wiki/White-Paper>, 2013.
  - [3] A. Back. Hashcash : a denial of service counter-measure. <http://www.hashcash.org/papers/hashcash.pdf>, 2002.
  - [4] J. Kwon and E. Buchman. Cosmos: a network of distributed ledgers. <https://github.com/cosmos/cosmos/blob/master/WHITEPAPER.md>, 2016.
  - [5] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation, Vol.8, No.1, January 1998, Pages 3-30.*, 1998.
  - [6] S. Popov. On a decentralized trustless pseudo-random number generation algorithm. <https://eprint.iacr.org/2016/228.pdf>, 2017.
  - [7] randao.org. Randao: verifiable random number generation. [https://randao.org/whitepaper/Randao\\_v0.85\\_en.pdf](https://randao.org/whitepaper/Randao_v0.85_en.pdf), 2017.