

트론(TRON)

해당 번역본은 이용자가 디지털자산 관련 정보를 편리하게 확인할 수 있도록
단순 참고용으로 번역된 것이며, 투자 권유를 목적으로 하지 않습니다.
그 내용이 정확하지 않을 수 있으므로 정확한 정보 습득을 위해서는
원문을 참고하시거나 원문 작성 주체 측에 문의하시기 바랍니다.
두나무는 해당 번역본 내용의 진실성 및 정확성을 보장하지 않고,
제공된 정보에 의한 투자 결과에 대하여 어떠한 책임도 지지 않으며,
투자 손실은 투자자에게 귀속됩니다.

TRON: 고급 탈중앙화된 블록체인 플랫폼

백서 버전: 2.1

TRON 프로토콜 버전: 4.8.0

목차

1 서론

1.1 비전

1.2 배경

1.3 역사

1.3.1 탈중앙화된 인터넷의 기원 (2017)

1.3.2 핵심 개발 (2018)

1.3.3 기능 확장 및 채택 확대 (2019 - 2021)

1.3.4 지속적 성장과 생태계 지배 (2022 - 현재)

1.3.5 미래 전망

1.4 용어

2 아키텍처

2.1 코어 레이어

2.2 스토리지 레이어

2.2.1 블록체인 스토리지

2.2.2 상태 스토리지

- 2.3 애플리케이션 레이어
- 2.4 프로토콜
 - 2.4.1 프로토콜 버퍼
 - 2.4.2 HTTP
- 2.5 TRON 가상 머신(TVM)

3 합의

- 3.1 위임 지분 증명(Delegated Proof of Stake, DPoS)

4 계정

- 4.1 유형
- 4.2 생성
- 4.3 구조

5 블록

- 5.1 블록 헤더
 - 5.1.1 원시 데이터
 - 5.1.2 증인 서명
 - 5.1.3 블록 ID
- 5.2 트랜잭션
 - 5.2.1 서명
 - 5.2.2 대역폭 모델
 - 5.2.3 수수료
 - 5.2.4 지분 증명으로서의 트랜잭션(TaPoS)
 - 5.2.5 트랜잭션 확인
 - 5.2.6 구조

6 TRON 가상 머신(TVM)

- 6.1 소개
- 6.2 워크플로우
- 6.3 성능
 - 6.3.1 경량 아키텍처
 - 6.3.2 견고성
 - 6.3.3 높은 호환성
 - 6.3.4 저비용

7 스마트 컨트랙트

- 7.1 소개
- 7.2 에너지 모델
- 7.3 배포
- 7.4 트리거 함수
- 7.5 TRON 솔리디티

8 토큰

- 8.1 TRC-10 토큰
- 8.2 TRC-20 토큰
- 8.3 그 이후

9 거버넌스

- 9.1 슈퍼 대표(Super Representative)
 - 9.1.1 일반 사항
 - 9.1.2 선거
 - 9.1.3 보상
- 9.2 위원회
 - 9.2.1 일반 사항
 - 9.2.2 동적 네트워크 매개변수
 - 9.2.3 제안 생성
 - 9.2.4 제안 투표
 - 9.2.5 제안 취소
- 9.3 구조

10 DApp 개발

- 10.1 API
- 10.2 네트워크
- 10.3 리소스

11 결론

부록 A: 용어

부록 B: 동적 네트워크 매개변수

1 서론

1.1 비전

TRON은 진정한 탈중앙화된 인터넷과 그 인프라를 구축하기 위해 헌신하는 야심찬 프로젝트이다. 세계에서 가장 큰 블록체인 기반 운영체제 중 하나인 TRON 프로토콜은 TRON 생태계 내 모든 탈중앙화된 애플리케이션(DApp)에 대해 높은 처리량, 높은 확장성, 높은 가용성을 갖춘 공용 블록체인 지원을 제공한다. 출범 이후 TRON은 꾸준히 기능을 확장하여 스테이블코인 송금과 같은 분야에서 상당한 채택을 달성하고 활발한 커뮤니티를 형성함으로써 블록체인 산업 내에서 그 두드러진 위치를 더욱 공고히 해왔다.

1.2 배경

2009년 비트코인의 도입은 글로벌 금융위기(2007-2008) 이후 전통적 금융 시스템에 대한 사회의 인식을 혁신적으로 바꾸었다. 중앙화된 헤지펀드와 은행들이 불투명한 금융 파생상품 투기로 인해 붕괴하는 가운데, 블록체인 기술은 누구나 트랜잭션 정보를 열람할 수 있는 투명한 범용 원장을 제공하였다. 트랜잭션은 작업 증명(Proof of Work, PoW) 합의 메커니즘을 통해 암호학적으로 보안되어 이중 지불 문제를 방지하였다.

2013년 말, 이더리움 백서는 스마트 컨트랙트와 튜링 완전(EVM, Ethereum Virtual Machine)을 갖춘 네트워크를 제안하였으며, 이를 통해 개발자들은 DApp을 통해 네트워크와 상호작용할 수 있게 되었다. 그러나 2017년 비트코인과 이더리움의 트랜잭션 처리량이 정점에 달하면서, 낮은 처리 속도와 높은 수수료로 인해 기존 상태의 비트코인과 이더리움은 광범위한 채택에 적합하지 않다는 점이 명확해졌다. 이에 따라 TRON은 이러한 긴급한 확장성 문제를 해결하기 위한 혁신적 솔루션으로 설립되었다.

1.3 역사

1.3.1 탈중앙화된 인터넷의 기원 (2017)

TRON의 여정은 2017년 7월 싱가포르에서 시작되었으며, 핵심 비전은 진정한 탈중앙화된 인터넷의 창조였다. 이 야심찬 프로젝트는 2017년 8월 성공적인 초기 코인 공개(ICO)를 통해 중요한 개발 자금을 확보함으로써 빠르게 추진력을 얻었다. 2017년 12월 오픈소스 프로토콜을 공개하며 투명성과 개방적 협업에 대한 초기 의지를 입증했고, 출범과 동시에 커뮤니티 참여를 초대하였다¹¹⁸.

1.3.2 핵심 개발 (2018)

2018년에는 급격한 기술적 발전이 이루어졌다. 3월의 Shasta 테스트넷 출시는 5월에 이루어진 핵심적인 메인넷 런칭(Odyssey 2.0)을 위한 기반을 마련하였으며, 이를 통해 TRON은 새로운 세대의 DApp을 지원할 수 있는 고성능 독립형 레이어-1 블록체인으로 자리매김했다. 네트워크 효율성과 탈중앙화된 거버넌스를 우선시하며, TRON은 위임 지분 증명(Delegated Proof-of-Stake, DPoS) 합의 메커니즘을 구현하고 2018년 6월 슈퍼 대표(SR)를 선출하였다. 또한 이 해는 7월 BitTorrent 인수를 통해 전략적 확장을 이룩함으로써 탈중앙화된 콘텐츠 배포 영역에서 TRON의 영향력과 잠재력을 크게 강화한 시기이기도 했다. 그해 후반인 10월에는 이더리움과 호환되는 TRON 가상 머신(TVM)이 출시되어 개발자 유입과 생태계 성장을 견인하는 주요 촉매 역할을 했다. 이를 뒷받침하며, 11월에 진행된 Odyssey-v3.2 업그레이드는 자원 위임 기능을 도입하여 네트워크 활용성을 최적화하였다. 2018년 12월까지 TRON의 급성장한 생태계는 100만 개 이상의 사용자 계정을 확보하였다. 2018년 전반에 걸쳐 도입된 TRC-10 및 TRC-20 토큰 표준은 자산 토큰화 및 DApp 혁신을 위한 핵심 인프라를 제공하였다.

1.1.1 백서 V1.0은 https://tron.network/static/doc/white_paper_v_1.0.pdf에서 확인할 수 있으며, 백서 V2.0은 https://tron.network/static/doc/white_paper_v_2.0.pdf에서 확인할 수 있다.

1.3.3 기능 확장 및 채택 확대 (2019 - 2021)

TRON은 2019년에 핵심 프로토콜 업그레이드를 통해 지속적인 진화를 이어갔다. 3월의 Odyssey-v3.5는 계정 보안성과 유연성을 강화했으며, 10월의 Odyssey-v3.6.5는 스테이킹과 거버넌스 메커니즘을 최적화하였다. 이 시기는 또한 TRON이 스테이블코인 송금 분야의 선도 네트워크로 부상한 중요한 시기였다. 2019년부터 TRC-20 USDT가 속도와 낮은 트랜잭션 수수료를 기반으로 광범위하게 채택되었으며, 이러한 효용성은 사용자 성장의 주요 동력이 되었다. 2020년 9월까지 플랫폼은 1,000만 개 이상의 계정을 확보하였다. 스테이블코인 생태계 내에서 TRON의 영향력은 TRC-20 USDT 발행량의 급속한 증가로 더욱 두드러졌으며, 2021년 초 100억을 돌파하고 2021년 5월에는 300억을 초과하였다.

1.3.4 지속적 성장과 생태계 지배 (2022 - 현재)

이 시기에는 사용자 채택과 스테이블코인 지배력 양 측면에서 지속적인 견조한 성장이 입증되었다. TRON의 사용자 기반은 빠르게 확장되어 2022년 6월에는 1억 명을 돌파하였고, 2025년 4월에는 3억 명을 초과하였다. 동시에 TRC-20 USDT 발행량도 꾸준히 증가하여 2025년 4월에는 700억을 초과하였으며, 이를 통해 TRON은 글로벌 스테이블코인 생태계 내 핵심 인프라로서의 지위를 공고히 하였다. 이러한 확장은 시장 역학에 대응하여 핵심 플랫폼의 안정성, 확장성, 성능을 향상시키고 경제 모델을 조정하려는 TRON의 지속적인 노력과 맞물려 있다.

이 시기의 주요 프로토콜 발전에는 일련의 필수 대규모 업그레이드가 포함된다. GreatVoyage-v4.7.0.1 (Aristotle) 버전은 Stake 2.0 메커니즘과 TVM을 위한 동적 에너지 모델을 도입하여 자원 관리와 실행 효율성을 강화하였다. 이후 GreatVoyage-v4.7.2 (Periander) 버전은 TRON의 네트워크 모듈을 Libp2p v1.2.0으로 업그레이드하여 P2P 통신과 블록 동기화를 개선하였다. 또한 Stake 2.0 및 TVM 내 EIP-3855 PUSH0 명령어에 대한 핵심 업데이트를 포함하였다. GreatVoyage-v4.8.0 (Kant) 버전은 이더리움의 최근 업그레이드에서 도입된 주요 개선사항을 반영하여 TRON 가상 머신(TVM)을 한층 강화하였으며, 스마트 컨트랙트 효율을 높이고 트랜잭션 비용을 줄여 EVM 호환 블록체인으로서의 TRON의 경쟁력을 유지하는 것을 목표로 한다.

1.3.5 미래 전망

확립된 인프라를 기반으로, TRON은 상호운용성 강화를 지속하고, 더 폭넓은 채택을 촉진하며, 탈중앙화 기술이 진화하는 환경 속에서 스테이블코인 전송 효율성과 활발한 생태계를 갖춘 선도적 블록체인으로서의 지위를 더욱 공고히 하는 데 전념하고 있다.

1.4 용어

이 백서 전반에서 사용되는 주요 용어와 약어의 정의는 부록 A: 용어를 참조하라.

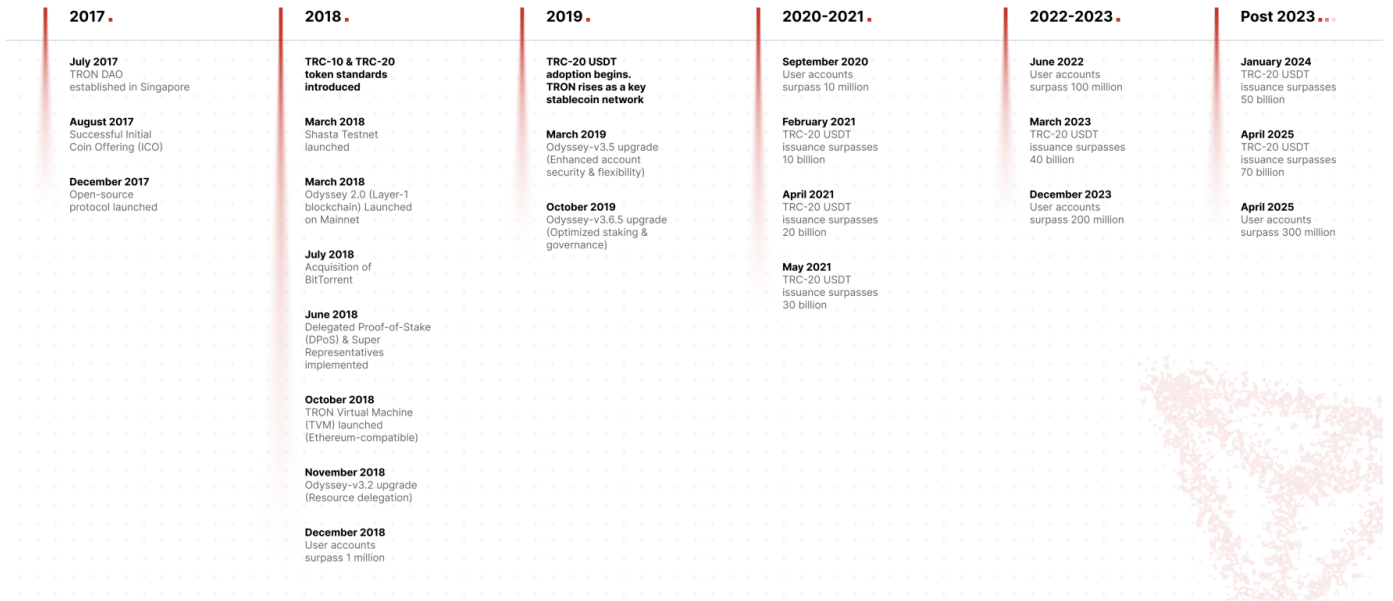


그림 1: TRON 개발 타임라인 및 주요 이정표 (2017–현재)

2 아키텍처

TRON은 코어 레이어(Core Layer), 스토리지 레이어(Storage Layer), 애플리케이션 레이어(Application Layer)로 구성된 3계층 아키텍처를 채택한다. TRON 프로토콜은 구글의 프로토콜 버퍼(Protocol Buffers, Protobuf)를 준수하며, 이는 본질적으로 다중 언어 확장을 지원한다.

2.1 코어 레이어

코어 레이어는 스마트 컨트랙트, 계정 관리, 합의 등 다양한 기능을 담당하는 여러 핵심 모듈로 구성된다. TRON은 스택 기반 가상 머신인 TRON 가상 머신(TVM)을 구현하며, 이는 최적화된 명령어 집합을 사용한다. DApp 개발자를 보다 잘 지원하기 위해 스마트 컨트랙트 언어로 Solidity²가 선택되었으며, 향후 다른 고급 언어에 대한 지원이 추가될 예정이다. 또한 TRON의 합의 메커니즘은 위임 지분 증명(Delegated Proof of Stake, DPoS)에 기반하며, 고유한 요구사항을 충족하기 위해 여러 혁신적 요소가 도입되었다.

2.2 스토리지 레이어

TRON의 고유한 탈중앙화된 스토리지 프로토콜은 블록 스토리지(Block Storage)와 상태 스토리지(State Storage)로 구성된다. 스토리지 레이어 설계에는 그래프 데이터베이스(Graph Database)의 개념이 도입되어, 현실 세계의 다양한 데이터 저장 요구를 더 잘 충족하도록 설계되었다.

2.2.1 블록체인 스토리지

TRON 블록체인 스토리지는 구글이 개발하였으며 다수의 기업과 프로젝트에서 검증된 LevelDB를 사용한다. 이는 높은 성능을 제공하며 키와 값을 임의의 바이트 배열로 지원하고, 개별 키에 대한 연산(get, put, delete), 일괄 연산(put, delete), 양방향 이터레이터, 그리고 빠른 Snappy 알고리즘을 활용한 단순 압축 기능을 제공한다.

2.2.2 상태 스토리지

TRON은 풀노드 메모리에 KhaosDB를 보유하고 있으며, 이는 일정 기간 내 생성된 모든 새로운 포크 체인을 저장할 수 있다. 이를 통해 슈퍼 대표(SR)는 자신이 운영 중인 활성 체인에서 새로운 메인 체인으로 신속하게 전환할 수 있다. 또한 KhaosDB는 블록체인 스토리지가 중간 상태에서 비정상적으로 종료되는 것을 방지하여 시스템의 안정성을 높인다.

^2^ Solidity 공식 문서: <https://solidity.readthedocs.io/>

2.3 애플리케이션 레이어

개발자는 TRON 위에서 다양한 DApp과 맞춤형 지갑을 개발할 수 있다. TRON은 스마트 컨트랙트의 배포와 실행을 지원하므로, 광범위한 잠재적 활용 사례를 지원한다.

2.4 프로토콜

TRON 프로토콜은 구글 프로토콜 버퍼(Google Protocol Buffers)^3^를 준수하며, 이는 통신 프로토콜, 데이터 저장 등에서 구조화된 데이터를 직렬화하기 위한 언어 중립적, 플랫폼 중립적, 확장 가능한 방식이다.

2.4.1 프로토콜 버퍼

프로토콜 버퍼(Protobuf)는 JSON이나 XML과 유사한 구조화된 데이터 직렬화를 위한 유연하고 효율적이며 자동화된 메커니즘으로, 훨씬 더 작고, 빠르며, 단순하다.

Protobuf(.proto) 정의는 공식 코드 생성기를 통해 C++, Java, C#, Python, Ruby, Golang, Objective-C 언어용 코드를 생성할 수 있다. 또한 다양한 서드파티 구현이 여러 언어로 제공된다.

Protobuf는 API 정의를 통합하고 데이터 전송을 최적화함으로써 클라이언트 개발을 단순화한다. 클라이언트는 TRON의 프로토콜 저장소에서 제공되는 API.proto를 사용하고 자동 생성된 코드 라이브러리를 통해 통합할 수 있다.

비교하자면, 프로토콜 버퍼는 XML보다 3~10배 작고 20~100배 빠르며, 문법적 모호성이 적다.

Protobuf는 프로그래밍적으로 더 쉽게 사용할 수 있는 데이터 접근 클래스를 생성한다.

2.4.2 HTTP

TRON 프로토콜은 Protobuf API의 대안으로 RESTful HTTP API를 제공한다. 두 API는 동일한 인터페이스를 공유하지만, HTTP API는 자바스크립트 클라이언트에서 보다 쉽게 사용할 수 있다.

2.5 TRON 가상 머신(TVM)

TVM은 TRON 생태계를 위해 개발된 경량의 튜링 완전(Turing-complete) 가상 머신이다. TVM은 기존 개발 생태계와 원활히 연동되어, 전 세계 수백만 명의 개발자에게 효율적이고 편리하며 안정적이고 안전하며 확장 가능한 맞춤형 블록체인 시스템을 제공한다.

^3^ 구글 프로토콜 버퍼 공식 문서: <https://developers.google.com/protocolbuffers/>

3 합의

3.1 위임 지분 증명(Delegated Proof of Stake, DPoS)

가장 초기의 합의 메커니즘은 작업 증명(Proof of Work, PoW) 합의 메커니즘이다. 이 프로토콜은 현재 비트코인^4^과 이더리움^5^에서 구현되어 있다. PoW 시스템에서는 네트워크를 통해 브로드캐스트된 트랜잭션이 채굴자 확인을 위한 새로운 블록으로 그룹화된다. 이 확인 과정은 암호학적 해시 알고리즘을 사용하여 트랜잭션을 해싱하고 머클 루트(merkle root)에 도달할 때까지 이를 반복함으로써 머클 트리(merkle tree)를 생성한다.

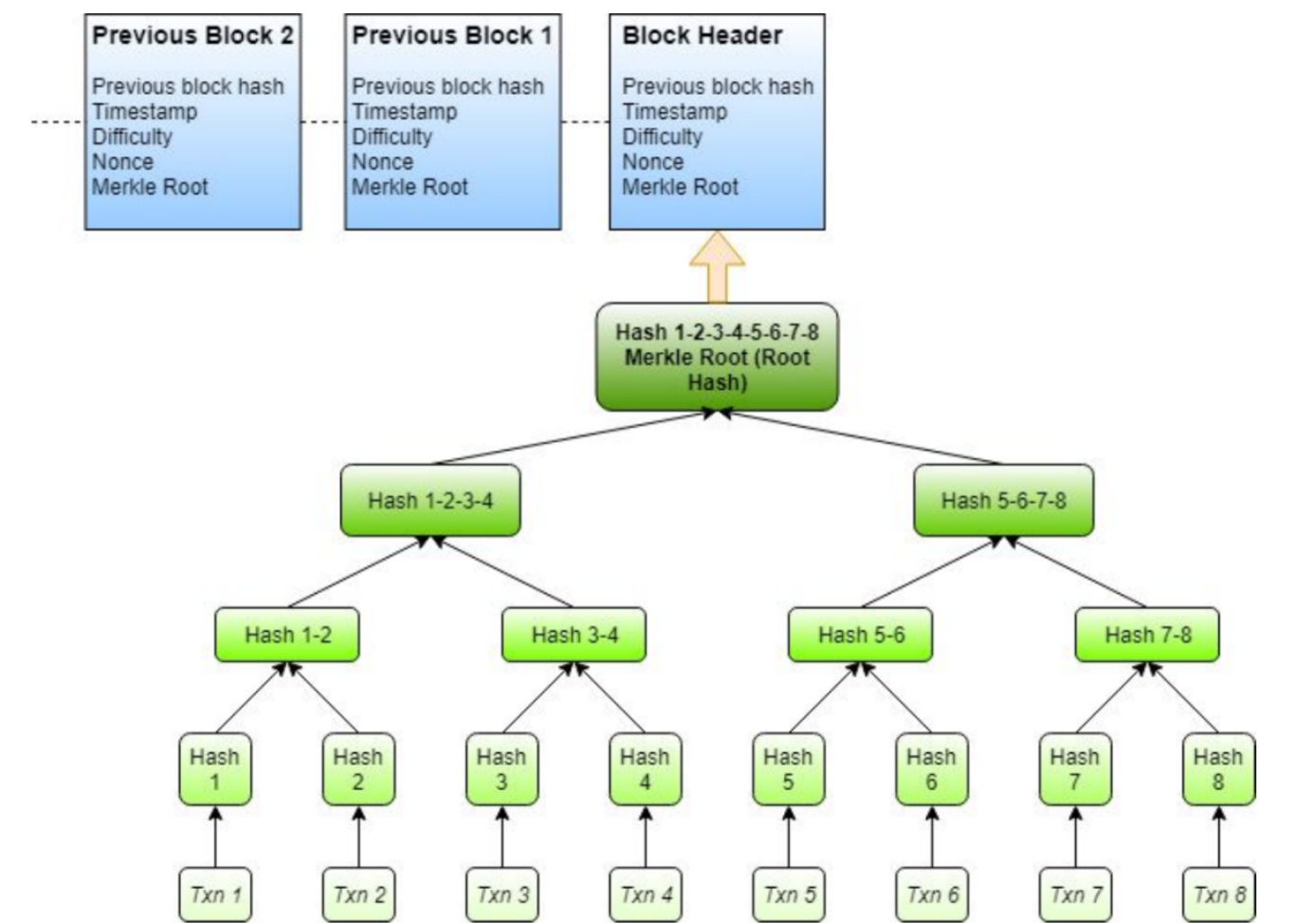


그림 3: 8개의 TRX 트랜잭션이 해시되어 머클 루트가 생성된다. 이 머클 루트는 블록 헤더에 포함되며, 이전에 확인된 블록에 연결되어 블록체인을 형성한다. 이를 통해 트랜잭션, 타임스탬프 및 기타 관련 정보를 쉽게 투명하게 추적할 수 있다.

암호학적 해시 알고리즘은 여러 속성을 가지고 있기 때문에 네트워크 공격 방지에 유용하다⁴⁸:

- **입력/출력 길이 크기** — 알고리즘은 임의의 길이를 가진 입력을 받을 수 있으며, 고정된 길이의 해시 값을 출력한다.
- **효율성** — 알고리즘은 비교적 계산이 쉽고 빠르다.
- **전상 이미지 저항성(Preimage resistance)** — 주어진 출력 z 에 대해 $h(x) = z$ 를 만족하는 입력 x 를 찾는 것은 불가능하다. 즉, 해시 알고리즘 $h(x)$ 는 출력만 알 수 있고 원래 입력을 찾을 수 없는 일방향 함수이다. 반대로 계산하는 것은 불가능하다.

⁴⁸ 비트코인 백서: <https://bitcoin.org/bitcoin.pdf>

⁴⁹ 이더리움 백서: <https://github.com/ethereum/wiki/wiki/WhitePaper>

⁵⁰ Paar, C., Pelzl, J., *Understanding Cryptography: A Textbook for Students and Practitioners*, 2010판, Springer-Verlag Berlin Heidelberg, 2010.

- **충돌 저항성(Collision resistance)** — $x_1 \neq x_2$ 인 쌍 중 $h(x_1) = h(x_2)$ 를 만족하는 값을 찾는 것은 계산적으로 불가능하다. 즉, 서로 다른 두 입력이 동일한 출력으로 해시될 확률은 극히 낮다. 이 속성은 또한 제2 전상 이미지 저항성(Second preimage resistance)을 내포한다.
- **제2 전상 이미지 저항성(Second preimage resistance)** — 주어진 x_1 과 그에 따른 $h(x_1)$ 이 있을 때, $h(x_1) = h(x_2)$ 를 만족하는 x_2 를 찾는 것은 계산적으로 불가능하다. 이 속성은 충돌 저항성과 유사하지만, 공격자가 주어진 x_1 에 대해 동일한 출력으로 해시되는 x_2 를 찾는 것이 계산적으로 불가능하다는 점에서 차이가 있다.
- **결정성(Deterministic)** — 각 입력은 단 하나의 출력에만 매핑된다.
- **눈사태 효과(Avalanche effect)** — 입력값이 조금만 변경되어도 완전히 다른 출력이 생성된다.

이러한 속성들은 네트워크가 공격에 의해 손상되지 않도록 보장함으로써 암호화폐 네트워크의 본질적 가치를 제공한다. 채굴자가 블록을 확인하면, 네트워크 참여에 대한 내장된 인센티브로서 토큰 보상을 받는다. 그러나 전 세계 암호화폐 시가총액이 꾸준히 증가함에 따라, 채굴자들은 점차 중앙화되고 네트워크 참여보다는 자산 축적을 목적으로 컴퓨팅 자원을 집중하게 되었다. CPU 채굴은 GPU로, 이후 강력한 ASIC으로 대체되었다.

에너지 낭비 문제를 해결하기 위해 많은 신규 네트워크가 지분 증명(Proof of Stake, PoS) 합의 메커니즘을 도입하였다. PoS 네트워크에서는 토큰 보유자가 자신의 토큰 잔액을 잠가 블록 검증자가 된다. 검증자들은 번갈아가며 다음 블록을 제안하고 투표한다. 그러나 일반적인 PoS의 문제점은 검증자의 영향력이 잠가둔 토큰의 양에 비례한다는 것이다. 그 결과, 네트워크의 기본 통화를 대량으로 보유한 일부 주체가 네트워크 생태계 내에서 과도한 영향력을 행사하게 된다.

TRON의 합의 메커니즘은 27명의 슈퍼 대표(SR, Super Representative)가 네트워크의 블록을 생성하는 혁신적인 위임 지분 증명(Delegated Proof of Stake, DPoS) 시스템을 사용한다. 매 6시간마다 계정을 동결한 TRX 보유자는 SR 후보에게 투표할 수 있으며, 상위 27명의 후보가 SR로 선정된다. 투표자는 TRX 채택을 촉진하기 위해 SR이 후원하는 프로젝트나 유권자에게 배분되는 보상 등의 기준에 따라 SR을 선택할 수 있다. 이를 통해 보다 민주적이고 탈중앙화된 생태계가 가능해진다. SR의 계정은 일반 계정이지만, 축적된 투표 수를 통해 블록을 생성할 권한을 얻게 된다. PoW 합의 메커니즘과 확장성 문제로 인해 낮은 처리 속도를 보이는 비트코인과 이더리움(비트코인 3 TPS, 이더리움 15 TPS)에 비해, TRON의 DPoS 시스템은 2000 TPS의 성능을 제공하는 혁신적 메커니즘을 제시한다.

TRON 프로토콜 네트워크는 3초마다 하나의 블록을 생성하며, 각 블록은 SR에게 16 TRX를 보상으로 지급한다. 매년 총 168,192,000 TRX가 27명의 SR에게 분배된다. 각 SR이 블록 생성을 완료할 때마다 보상은 슈퍼 원장(Super Ledger)의 하위 계정으로 전송된다. SR은 해당 TRX를 확인할 수는 있으나 직접 사용할 수는 없으며, 각 SR은 24시간마다 한 번씩 하위 계정에서 지정된 SR 계정으로 보상을 인출할 수 있다.

TRON 네트워크의 노드는 세 가지 유형으로 구성된다: 증인 노드(Witness Node), 풀 노드(Full Node), 라이트 풀 노드(Lite Full Node). 증인 노드는 SR에 의해 설정되며, 블록 생성 및 제안 작성/투표를 주로 담당한다. 풀 노드는 API를 제공하고 트랜잭션 및 블록을 브로드캐스트한다. 솔리디티 노드는 다른 풀 노드로부터 블록을 동기화하며 인덱싱 가능한 API도 제공한다.

4 계정

4.1 유형

TRON 네트워크에는 두 가지 주요 유형의 계정이 있다.

1. **외부 소유 계정(Externally Owned Accounts, EOA)**: 개인 키로 제어되는 계정이다. TRX 및 토큰 전송, 슈퍼 대표(SR) 투표, 스마트 컨트랙트 배포에 사용된다. EOA 계정은 TRX, TRC-10 토큰, TRC-20/TRC-721 토큰(컨트랙트 계정과의 상호작용을 통해)을 보유할 수 있다. 이는 모든 지갑 사용자가 사용하는 가장 일반적인 유형의 계정이다.
2. **컨트랙트 계정(Contract Accounts)**: 코드의 로직에 의해 제어되는 스마트 컨트랙트 계정이다. 컨트랙트 계정에는 개인 키가 없으며 생성과 동시에 활성화된다. 해당 계정의 기능은 코드에서 허용하는 범위 내에서 EOA 또는 다른 컨트랙트에 의해 호출될 수 있다.

4.2 생성

TRON 계정은 다음 세 가지 방법으로 생성할 수 있다.

1. 명령줄 지갑 `wallet-cli`를 사용하여 오프라인으로 계정을 생성한다.
2. Trident SDK와 같은 SDK를 사용하여 오프라인으로 계정을 생성한다.
3. 지갑 애플리케이션을 통해 개인 키 및 해당 주소를 생성한다.

계정은 다음 두 가지 방식으로 활성화할 수 있다.

1. 기존 계정에서 새 계정으로 임의의 양의 TRX 또는 TRC-10 토큰을 전송한다.
2. Java-tron의 `wallet/createaccount` API를 호출하여 기존 계정에서 트랜잭션을 생성한 후, 해당 트랜잭션에 서명하고 TRON 네트워크에 브로드캐스트한다.

TRON 네트워크에 기록되지 않은 주소(공개 키)와 개인 키로 구성된 오프라인 키 쌍도 생성할 수 있다. 사용자 주소 생성 알고리즘은 키 쌍을 생성한 후 공개 키(64바이트 배열, x, y 좌표로 구성)를 추출하는 과정으로 시작한다. 공개 키를 SHA3-256 함수(채택된 SHA3 프로토콜은 KECCAK-256)를 사용해 해시하고 결과의 마지막 20바이트를 추출한다. 바이트 배열의 시작에 41을 추가하여 초기 주소 길이를 21바이트로 설정한다. 그런 다음 SHA3-256 함수를 사용해 주소를 두 번 해시하고 첫 4바이트를 검증 코드로 취한다. 검증 코드를 초기 주소의 끝에 추가하고 base58 인코딩을 통해 base58check 형식의 주소를 얻는다. 인코딩된 메인넷 주소는 T로 시작하며 34바이트 길이를 가진다.

4.3 구조

TRON의 세 가지 계정 유형은 Normal, AssetIssue, Contract이다. 하나의 Account는 다음 7개의 매개변수를 포함한다.

- **account_name**: 계정의 이름 — 예: BillsAccount.
- **type**: 계정 유형 — 예: 0 (유형 'Normal'을 의미).
- **address**: TRON 네트워크 내에서 계정을 식별하는 고유 식별자 — 예: T9yD...9mGg.
- **balance**: 계정의 TRX 잔액 — 예: 4213312.
- **vote**: 계정이 받은 투표 수 — 예: {("0x1b7w...9xj3", 323), ("0x8dj...j12m", 88), ..., ("0x82nd...mxi6", 10001)}.
- **asset**: TRX 이외의 자산 — 예: {<"WishToken", 66666>, <"Dogie", 233>}.
- **latest operation time**: 계정의 최근 작업에 대한 타임스탬프.

프로토콜 버퍼(Protobuf) 데이터 구조:

```

1 message Account {
2     message Vote {
3         bytes vote_address = 1;
4         int64 vote_count = 2;
5     }
6     bytes account_name = 1;
7     AccountType type = 2;
8     bytes address = 3;
9     int64 balance = 4;
10    repeated Vote votes = 5;
11    map<string, int64> asset = 6;
12    int64 latest_operation_time = 10;
13 }
14 enum AccountType {
15     Normal = 0;
16     AssetIssue = 1;
17     Contract = 2;
18 }

```

5 블록

블록은 일반적으로 블록 헤더와 여러 트랜잭션으로 구성된다.
프로토콜 버퍼(Protobuf) 데이터 구조:

```
message Block {  
    BlockHeader block_header = 1;  
    repeated Transaction transactions = 2;  
}
```

5.1 블록 헤더

블록 헤더는 raw_data, witness_signature, blockID로 구성된다.
프로토콜 버퍼(Protobuf) 데이터 구조:

```
message BlockHeader {  
    message raw {  
        int64 timestamp = 1;  
        bytes txTrieRoot = 2;  
        bytes parentHash = 3;  
        int64 number = 7;  
        int64 witness_id = 8;  
        bytes witness_address = 9;  
        int32 version = 10;  
        bytes accountStateRoot = 11;  
    }  
    raw raw_data = 1;  
    bytes witness_signature = 2;  
}
```

5.1.1 원시 데이터(Raw Data)

원시 데이터는 Protobuf에서 raw_data로 표시되며, 메시지의 원시 데이터를 포함하며 다음 8개의 매개 변수를 가진다.

1. **timestamp**: 이 메시지의 타임스탬프 — 예: 1543884429000.
2. **txTrieRoot**: 머클 트리의 루트 — 예: 7dacs...3ed.
3. **parentHash**: 이전 블록의 해시 — 예: 7dacs...3ed.
4. **number**: 블록 높이 — 예: 4638708.
5. **version**: 예약된 값 — 예: 5.
6. **witness_address**: 이 블록을 패킹한 증인의 주소 — 예: 41928c...4d21.
7. **witness_id**: 이 블록을 패킹한 증인의 ID — 예: 0xu82h...7237.
8. **accountStateRoot**: 이 블록에 패킹된 증인의 주소 — 예: 0xu82h...7237.

5.1.2 증인 서명(Witness Signature)

증인 서명은 Protobuf에서 witness_signature로 표시되며, 증인 노드가 이 블록 헤더에 대해 생성한 서명이다.

5.1.3 블록 ID(Block ID)

블록 ID는 Protobuf에서 blockID로 표시되며, 블록의 고유 식별 정보를 포함한다. 하나의 블록 ID는 두 개의 매개변수로 구성된다.

1. **hash**: 블록의 해시.
2. **number**: 블록의 해시 및 높이.

5.2 트랜잭션

5.2.1 서명(Signing)

TRON의 트랜잭션 서명 과정은 표준 ECDSA 암호화 알고리즘을 따르며, SECP256K1 선택 곡선을 사용한다. 개인 키는 임의의 숫자이며, 공개 키는 타원 곡선상의 한 점이다. 공개 키 생성 과정은 먼저 임의의 숫자를 개인 키로 생성한 다음, 타원 곡선의 기준점을 개인 키로 곱하여 공개 키를 얻는 방식으로 이루어진다. 트랜잭션이 발생하면, 트랜잭션의 원시 데이터(raw data)는 먼저 바이트 형식으로 변환된다.

그 후 원시 데이터는 SHA-256 해싱 과정을 거친다. 해당 계약 주소에 대응하는 개인 키가 SHA256 해시 결과를 서명하며, 서명 결과는 트랜잭션에 추가된다.

5.2.2 대역폭 모델(Bandwidth Model)

일반 트랜잭션은 대역폭 포인트(Bandwidth Points)만 소비하지만, 스마트 컨트랙트 작업은 에너지(Energy)와 대역폭 포인트를 모두 소비한다. 대역폭 포인트는 두 가지 방식으로 얻을 수 있다. 사용자는 TRX를 동결하여 대역폭 포인트를 얻을 수 있으며, 매일 무료 대역폭 포인트도 제공된다⁷⁸.

TRX 트랜잭션이 브로드캐스트되면 네트워크를 통해 바이트 배열 형태로 전송 및 저장된다. 한 트랜잭션이 소비하는 대역폭 포인트 = 트랜잭션 바이트 수 × 대역폭 포인트 비율. 예를 들어, 트랜잭션의 바이트 배열 길이가 200이라면, 해당 트랜잭션은 200 대역폭 포인트를 소비한다. 그러나 TRX 또는 토큰 전송이 대상 계정을 새로 생성하는 경우, 계정 생성에 필요한 대역폭 포인트만 차감되며 추가적인 대역폭 포인트는 차감되지 않는다. 계정 생성 시 충분한 대역폭이 없는 경우, 제안(proposal)에 의해 제어되는 계정 생성 수수료 0.1 TRX가 차감된다. 계정 생성 트랜잭션의 크기 제한은 현재 1000바이트이며, 이 역시 제안에 의해 관리된다. 계정 생성 상황에서는 네트워크가 먼저 트랜잭션 생성자가 TRX를 동결하여 얻은 대역폭 포인트를 사용한다. 만약 부족할 경우, 네트워크는 트랜잭션 생성자의 TRX를 소비한다.

TRX 계정 간의 일반적인 전송 시나리오에서는 네트워크가 먼저 트랜잭션 생성자가 TRX를 동결하여 얻은 대역폭 포인트를 소비한다. 부족할 경우, 매일 제공되는 무료 대역폭 포인트를 사용하고, 여전히 부족할 경우 네트워크는 트랜잭션 생성자의 TRX를 소비한다. 이때 소비되는 양은 트랜잭션의 바이트 수 × 바이트당 TRX 비용⁸⁸으로 계산된다. 따라서 SR 투표에 참여하기 위해 TRX를 동결하지 않은 일반 TRX 보유자의 경우, 첫 단계(동결 TRX 사용)는 자동으로 건너뛰며, 무료 대역폭 포인트가 트랜잭션을 처리하게 된다.

TRC-10 토큰 전송의 경우, 네트워크는 먼저 발행된 토큰 자산의 전체 무료 대역폭 포인트가 충분한지 확인한다. 부족할 경우, 동결된 TRX에서 얻은 대역폭 포인트를 소비한다. 여전히 부족하다면, 네트워크는 트랜잭션 생성자의 TRX를 소비한다.

5.2.3 수수료(Fee)

TRON 네트워크는 대부분의 트랜잭션에 대해 일반적으로 수수료를 부과하지 않지만, 시스템 제약과 공정성을 위해 대역폭 사용 및 트랜잭션에는 일정한 수수료가 발생한다.

수수료는 다음과 같이 구분된다:

1. **일반 트랜잭션**은 대역폭 포인트를 소비한다. 사용자는 무료 일일 대역폭 포인트를 사용하거나 TRX를 동결하여 추가 대역폭을 얻을 수 있다. 대역폭 포인트가 부족할 경우, 송신 계정의 TRX가 직접 사용된다. 필요한 TRX는 '바이트 수 × 바이트당 TRX 비용'으로 계산된다.
2. **스마트 컨트랙트**는 에너지(Energy)를 소비하며(자세한 내용은 [6장](#) 참조), 트랜잭션의 브로드캐스트 및 확인을 위해 대역폭 포인트도 필요하다. 대역폭 비용은 위와 동일하게 계산된다.

Λ7Λ 무료 대역폭 포인트의 구체적인 한도는 TRON 체인 매개변수 API의 freeNetLimit 파라미터로 정의된다: <https://developers.tron.network/reference/wallet-getchainparameters>
Λ8Λ 바이트당 TRX 비용은 TRON 체인 매개변수 API의 getTransactionFee 파라미터로 정의된다.

3. 모든 조회 트랜잭션(Query Transactions)은 무료이다. 에너지(Energy)나 대역폭(Bandwidth)을 소비하지 않는다.

TRON 네트워크는 또한 다음 트랜잭션에 대해 고정 수수료를 정의한다(최신 값은 TRON 체인 매개변수 API를 참조):

1. **TRC-10 토큰 발행**: 1,024 TRX (getAssetIssueFee 참조)
2. **SR 후보 등록 신청**: 9,999 TRX (getAccountUpgradeCost 참조)
3. **Bancor 트랜잭션 생성**: 1,024 TRX (getExchangeCreateFee 참조)
4. **계정 권한 업데이트**: 100 TRX (getUpdateAccountPermissionFee 참조)
5. **계정 활성화**: 1 TRX (getCreateNewAccountFeeInSystemContract 참조); 생성자의 계정에서 스테이킹을 통해 확보된 대역폭이 부족한 경우, 대역폭 비용으로 추가 0.1 TRX가 부과된다 (getCreateAccountFee 참조)
6. **멀티시그(Multisig) 트랜잭션**: 1 TRX (getMultiSignFee 참조)
7. **트랜잭션 메모(Transaction Note)**: 1 TRX (getMemoFee 참조)

5.2.4 지분 증명으로서의 트랜잭션(Transaction as Proof of Stake, TaPoS)

TRON은 모든 트랜잭션이 메인 블록체인을 확인하도록 보장하고 위조된 체인을 만들기 어렵게 하기 위해 TaPoS를 사용한다. TaPoS에서는 네트워크가 각 트랜잭션에 최근 블록 헤더의 일부 해시를 포함하도록 요구한다. 이 요구사항은 참조된 블록을 포함하지 않는 포크에서 트랜잭션이 재생(replay)되는 것을 방지하며, 특정 사용자와 그들의 스테이크가 특정 포크에 속해 있음을 네트워크에 신호로 전달한다. 이 합의 메커니즘은 서비스 거부(DoS), 51% 공격, 이기적 채굴(Selfish Mining), 이중 지출(Double-Spending)

공격으로부터 네트워크를 보호한다.

5.2.5 트랜잭션 확인(Transaction Confirmation)

트랜잭션은 네트워크에 브로드캐스트된 후 블록에 포함된다. 해당 블록이 서로 다른 슈퍼 대표(SR)에 의해 생성된 19개의 연속 블록에 의해 뒤따를 때, 트랜잭션은 확인된 것으로 간주된다.

각 블록은 블록체인 상에서 약 3초마다 채굴되며, 네트워크 상태 및 장비 구성에 따라 SR별로 시간이 약간 다를 수 있다. 일반적으로 트랜잭션은 약 1분 후 완전히 확인된 것으로 간주된다.

5.2.6 구조(Structure)

트랜잭션 API는 다음과 같은 기능으로 구성된다:

```

1 message Transaction {
2     message Contract {
3         enum ContractType {
4             AccountCreateContract = 0; // Create account/wallet
5             TransferContract = 1; // Transfer TRX
6             TransferAssetContract = 2; // Transfer TRC-10 token
7             VoteWitnessContract = 4; // Vote for SR
8             WitnessCreateContract = 5; // Create a new SR account
9             AssetIssueContract = 6; // Create a new TRC-10 token
10            WitnessUpdateContract = 8; // Update SR information
11            ParticipateAssetIssueContract = 9; // Purchase TRC-10 token
12            AccountUpdateContract = 10; // Update account/wallet information
13            FreezeBalanceContract = 11; // Freeze TRX for Bandwidth or Energy
14            UnfreezeBalanceContract = 12; // Unfreeze TRX
15            WithdrawBalanceContract = 13; // Withdraw SR rewards, once per day
16            UnfreezeAssetContract = 14; // Unfreeze TRC-10 token
17            UpdateAssetContract = 15; // Update a TRC-10 token's information
18            ProposalCreateContract = 16; // Create a new network proposal by any SR
19            ProposalApproveContract = 17; // SR votes yes for a network proposal
20            ProposalDeleteContract = 18; // Delete a network proposal by owner

21            CreateSmartContract = 30; // Deploy a new smart contract
22            TriggerSmartContract = 31; // Call a function on a smart contract
23            GetContract = 32; // Get an existing smart contract
24            UpdateSettingContract = 33; // Update a smart contract's parameters
25            ExchangeCreateContract = 41; // Create a token trading pair on DEX
26            ExchangeInjectContract = 42; // Inject funding into a trading pair
27            ExchangeWithdrawContract = 43; // Withdraw funding from a trading pair
28            ExchangeTransactionContract = 44; // Perform token trading
29            UpdateEnergyLimitContract = 45; // Update origin_energy_limit on a smart
contract
30            AccountPermissionUpdateContract = 46; // Update account permissions
31            ClearABIContract = 48; // Clear a smart contract's ABI
32            UpdateBrokerageContract = 49; // Update SR brokerage commission rate
33            MarketSellAssetContract = 52; // Place a market sell order
34            MarketCancelOrderContract = 53; // Cancel an existing market order
35            FreezeBalanceV2Contract = 54; // Freeze TRX to get resources
36            UnfreezeBalanceV2Contract = 55; // Cancel freezed TRX
37            WithdrawExpireUnfreezeContract = 56; // Withdraw TRX after the unfreeze waiting
period has expired
38            DelegateResourceContract = 57; // Delegate Bandwidth or Energy to another
account
39            UnDelegateResourceContract = 58; // Cancel delegated resource
40            CancelAllUnfreezeV2Contract = 59; // Cancel all pending unfreeze operations
41        }
42        ContractType type = 1;
43        google.protobuf.Any parameter = 2;
44        bytes provider = 3;
45        bytes ContractName = 4;
46        int32 Permission_id = 5;
47    }
48 }

```

6 TRON 가상 머신(TVM)

6.1 소개(Introduction)

TRON 가상 머신(TVM)은 TRON 생태계를 위해 개발된 경량의 튜링 완전(Turing-complete) 가상 머신이다. TVM의 목표는 효율적이고 편리하며 안정적이고 안전하며 확장 가능한 맞춤형 블록체인 시스템을 제공하는 것이다.

TVM은 초기에는 EVM⁹⁾에서 포크되어 시작했으며, 기존의 Solidity 스마트 컨트랙트 개발 생태계와 원활하게 연동될 수 있다. 이를 기반으로 TVM은 추가적으로 DPoS 합의 메커니즘을 지원한다.

TVM은 계산 자원을 관리하기 위해 에너지(Energy) 개념을 도입하였으며, 이는 EVM의 가스(Gas) 메커니즘과 다르다. TVM에서 스마트 컨트랙트의 실행은 에너지를 소비하며, 사용자는 TRX를 스테이킹하여 에너지를 획득할 수 있다. 계정에 충분한 에너지가 있을 경우, 스마트 컨트랙트 실행 시 TRX를 직접 소각할 필요가 없다. 그러나 사용 가능한 에너지가 부족할 경우, 시스템은 해당 계정의 TRX를 소각하여 계산 비용을 충당한다.

6.2 워크플로우(Workflow)

컴파일러는 먼저 Solidity 스마트 컨트랙트를 TVM에서 읽고 실행할 수 있는 바이트코드로 변환한다. 이후 TVM은 오퍼코드(opcode)를 통해 데이터를 처리하며, 이는 스택 기반 유한 상태 머신의 로직을 운영하는 것과 동일하다. 마지막으로 TVM은 블록체인 데이터를 접근하고 상호운용(Interoperation) 계층을 통해 외부 데이터 인터페이스를 호출한다.

6.3 성능(Performance)

6.3.1 경량 아키텍처(Lightweight Architecture)

TVM은 시스템 성능을 보장하기 위해 자원 소비를 줄이는 경량 아키텍처를 채택하였다.

6.3.2 견고성(Robust)

TRON은 트랜잭션 처리와 스마트 컨트랙트 실행을 위해 대역폭(Bandwidth)과 에너지(Energy)로 구성된 이중 자원 모델을 채택한다. 사용자는 TRX를 동결함으로써 이러한 자원을 직접적인 트랜잭션 비용 없이 획득할 수 있다. 자원 한도를 초과할 경우, 초과 사용량에 비례하여 TRX가 소각된다. 이 자원 관리 시스템은 악의적 활동의 경제적 비용을 높여 네트워크 보안을 강화하고, 각 작업 단위당 고정된 비용 구조로 예측 가능한 자원 소비를 보장한다.

6.3.3 높은 호환성(High Compatibility)

TVM은 EVM과 호환되며, 향후 더 많은 주류 가상 머신(VM)들과의 호환성도 확보할 예정이다. 따라서 EVM 상의 모든 스마트 컨트랙트는 TVM에서도 실행 가능하다.

6.3.4 저비용(Low Cost)

TVM의 대역폭 설계로 인해 개발 비용이 감소하며, 개발자는 컨트랙트 코드의 로직 개발에 집중할 수 있다. 또한 TVM은 컨트랙트 배포, 트리거, 조회를 위한 통합 인터페이스를 제공하여 개발자의 편의성을 높인다.

⁹⁾ EVM: Ethereum Virtual Machine (<https://github.com/ethereum/ethereumj>)

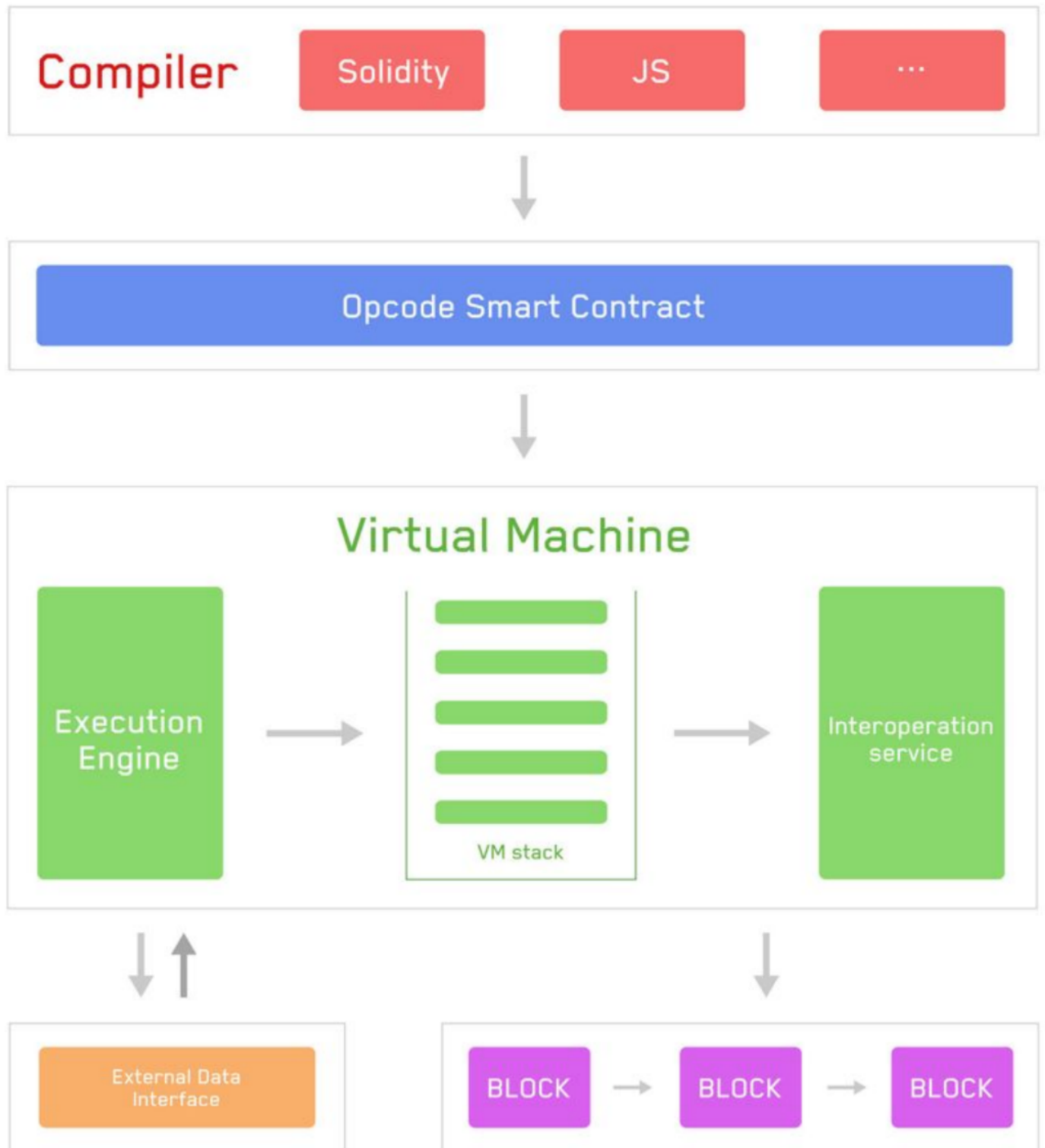


Figure 4: TVM 워크플로우

7 스마트 컨트랙트(Smart Contract)

7.1 소개(Introduction)

스마트 컨트랙트는 계약 협상을 디지털 방식으로 검증하는 프로토콜이다. 스마트 컨트랙트는 계약과 관련된 규칙과 위반 시 제재를 정의하며, 해당 의무를 자동으로 이행한다. 스마트 컨트랙트 코드는 계약이나 트랜잭션의 협상 또는 이행을 촉진하고, 검증하며, 집행한다. 토큰화 관점에서 볼 때, 스마트 컨트랙트는 특정 조건이 충족될 경우 참여자 간의 자동 자금 이체를 가능하게 한다.

TRON 스마트 컨트랙트는 Solidity 언어로 작성된다. 작성 및 테스트가 완료되면 바이트코드로 컴파일되어 TRON 가상 머신(TVM)을 통해 TRON 네트워크에 배포될 수 있다. 배포 후에는 컨트랙트 주소를 통해 스마트 컨트랙트를 조회할 수 있다. 컨트랙트 애플리케이션 바이너리 인터페이스(ABI)는 컨트랙트의 호출 함수 목록을 보여주며, 네트워크와 상호작용할 때 사용된다.

7.2 에너지 모델(Energy Model)

스마트 컨트랙트를 배포하고 트리거할 때의 최대 에너지 한도는 여러 변수의 함수로 결정된다.

- 1 TRX 동결로부터의 동적 에너지(Dynamic Energy): $180,000,000,000$ (총 에너지 한도) / (총 에너지 가중치)
- 에너지 한도: TRX 동결로부터 얻는 일일 계정 에너지 한도
- 남은 일일 계정 에너지: 에너지 한도 - 사용된 에너지
- TRX 단위의 수수료 한도: 스마트 컨트랙트 배포/트리거 호출 시 설정됨
- 계정 내 남은 사용 가능한 TRX
- TRX 직접 구매 시 에너지 비율($210 \text{ SUN} = 1 \text{ Energy}$) = $100,000$, SR 투표로 조정 가능

배포 및 트리거의 최대 에너지 한도를 계산하기 위한 두 가지 소비 시나리오가 있으며, 그 로직은 다음과 같이 표현된다.

```
1  const R = Dynamic Energy Limit
2  const F = Daily account Energy from freezing TRX
3  const E = Remaining daily account Energy from freezing TRX
4  const L = Fee limit in TRX set in deploy/trigger call
5  const T = Remaining usable TRX in account
6  const C = Energy per TRX if purchased directly
7  // Calculate M, defined as maximum Energy limit for deployment/trigger of a smart contract
8  if F > L*R
9      let M = min(E+T*C, L*R)
10 else
11     let M = E+T*C
```

2023년 2월 5일부터 TRON 네트워크는 TRON 개발자 커뮤니티가 제안한 제83호 위원회 제안을 통해 새로운 동적 에너지 모델을 도입하였다. 이 모델은 각 컨트랙트의 자원 점유율에 따라 에너지 소비를 동적으로 조정하며, 체인상의 에너지 자원 분배를 보다 합리적으로 만들고 일부 인기 컨트랙트에 네트워크 자원이 과도하게 집중되는 것을 방지하는 것을 목표로 한다.

7.3 배포(Deployment)

TRON Solidity 스마트 컨트랙트가 컴파일되면 TRON 가상 머신(TVM)은 컴파일된 바이트코드를 읽는다. 바이트코드는 코드 배포 섹션, 컨트랙트 코드, Auxdata 섹션으로 구성된다. Auxdata는 소스 코드의 암호학적 지문으로, 검증에 사용된다. 배포용 바이트코드는 생성자(Constructor) 함수를 실행하고 초기 스토리지 변수를 설정한다. 배포 코드는 또한 컨트랙트 코드를 계산하여 TVM에 반환한다. ABI는 TRON 스마트 컨트랙트의 기능을 설명하는 JSON 파일이며, 함수 이름, 결제 가능 여부(payability), 반환값, 상태 변경성(state mutability)을 정의한다.

7.4 트리거 함수(Trigger Function)

TRON 스마트 컨트랙트가 배포된 후에는 각 함수가 Tronide 또는 API 호출을 통해 개별적으로 트리거될 수 있다. 상태를 변경하는 함수는 에너지를 필요로 하며, 읽기 전용 함수는 에너지 소비 없이 실행된다.

7.5 TRON 솔리디티(TRON Solidity)

TRON Solidity는 이더리움의 Solidity 언어를 포크한 것이다. TRON은 원본 프로젝트를 수정하여 TRX 및 SUN 단위(1 TRX = 1,000,000 SUN)를 지원하도록 하였다. 나머지 언어 문법은 Solidity 0.8.23 버전과 호환된다. 따라서 TRON 가상 머신(TVM)은 EVM 명령어와 거의 100% 호환된다.

8 토큰(Token)

8.1 TRC-10 토큰

TRON 네트워크에서 각 계정은 1,024 TRX를 지불하여 토큰을 발행할 수 있다. 토큰을 발행하기 위해 발행자는 토큰 이름, 총 발행량, TRX와의 환율, 유통 기간, 설명, 웹사이트, 계정당 최대 대역폭 소비량, 전체 네트워크의 총 대역폭 소비량, 그리고 동결된 토큰의 양을 지정해야 한다. 또한 각 토큰 발행 시 계정당 일일 최대 토큰 전송 대역폭 포인트, 네트워크 전체의 일일 최대 토큰 전송 대역폭 포인트, 총 토큰 공급량, 잠금 기간(일 단위), 잠금된 토큰의 총량을 설정할 수 있다.

8.2 TRC-20 토큰

TRC-20은 TRON 가상 머신(TVM)이 지원하는 토큰을 구현하기 위한 스마트 컨트랙트 기술 표준으로, ERC-20과 완벽히 호환된다.

인터페이스는 다음과 같다:

```
1 contract TRC20Interface {
2     function totalSupply() public constant returns (uint);
3     function balanceOf(address tokenOwner) public constant returns (uint balance);
4     function allowance(address tokenOwner, address spender) public constant returns (uint
    remaining);
5     function transfer(address to, uint tokens) public returns (bool success); function
    approve(address spender, uint tokens) public returns (bool success);
6     function transferFrom(address from, address to, uint tokens) public returns (bool
    success);
7     event Transfer(address indexed from, address indexed to, uint tokens); event Approval(
    address indexed tokenOwner, address indexed spender, uint tokens);
8 }
```

TRC-10 토큰의 전송은 주로 대역폭(Bandwidth)을 소비하는 반면, TRC-20 토큰 전송은 대역폭과 에너지(Energy)를 모두 소비한다. 이러한 자원 소비 차이로 인해 TRC-10 토큰의 트랜잭션 수수료는 일반적으로 TRC-20보다 현저히 낮다. API 전송 및 입금 시 TRC-10 토큰은 대역폭 비용이 발생하지만, TRC-20 트랜잭션의 대역폭 및 에너지 소비를 합친 것보다 낮은 수준을 유지한다. TRC-10 토큰이 스마트 컨트랙트를 통해 전송되거나 입금될 경우 대역폭과 에너지가 모두 소모된다. 반면 TRC-20은 API 호출이든 스마트 컨트랙트 실행이든 관계없이 전송 및 입금 시 항상 대역폭과 에너지를 동시에 소비한다.

8.3 그 이후(Beyond)

TRON은 이더리움과 동일한 Solidity 버전을 사용하므로, 더 많은 토큰 표준을 손쉽게 TRON으로 이식할 수 있다.

9 거버넌스(Governance)

9.1 슈퍼 대표(Super Representative)

9.1.1 일반 사항(General)

TRON 네트워크의 모든 계정은 슈퍼 대표(SR, Super Representative)가 되기 위해 지원할 수 있으며, 누구나 SR 후보에게 투표할 수 있다. 가장 많은 표를 얻은 상위 27명의 후보가 블록을 생성할 권리와 의무를 가진 SR로 선출된다. 투표는 6시간마다 집계되며, 이에 따라 SR 구성도 변경된다.

악의적인 공격을 방지하기 위해 SR 후보로 등록하는 데에는 일정 비용이 필요하다. 지원 시 신청자의 계정에서 9,999 TRX가 소각되며, 이후 해당 계정은 SR 선거에 참여할 자격을 얻게 된다.

9.1.2 선거(Election)

투표에는 TRON 파워(TRON Power, TP)가 필요하며, TP의 양은 투표자의 동결된 자산(TRX)에 따라 결정된다.

TP는 다음과 같이 계산된다.

$$1TP = 1TRX_{frozen\ to\ get\ Energy/Bandwidth}$$

TRON 네트워크의 모든 계정은 자신이 선택한 SR에게 투표할 권리가 있다.

TRX가 해제(unfreeze)되면(14일 후 가능), 사용자는 더 이상 동결된 자산을 보유하지 않게 되며, 그에 따라 모든 TP를 잃게 된다. 그 결과, 현재 및 향후 투표 라운드에서 해당 사용자의 투표는 모두 무효가 되며, 다시 투표하려면 TRX를 재동결해야 한다.

TRON 네트워크는 오직 가장 최근의 투표만 기록하므로, 새로운 투표를 할 때마다 이전의 모든 투표는 무효화된다는 점에 유의해야 한다.

9.1.3 보상(Reward)

a. 투표 보상(Vote Reward)

투표 보상은 후보 보상(Candidate Reward)이라고도 하며, 매 라운드(6시간)마다 상위 127명의 후보가 총 921,600 TRX를 채굴된 형태로 공유한다. 보상은 각 후보가 받은 투표 비중에 따라 분배된다. 연간 후보자 총 보상액은 1,345,536,000 TRX이다.

라운드당 총 투표 보상(Total vote reward per round)

왜 매 라운드마다 921,600 TRX인가?

$$921,600 \text{ TRX} = \text{total vote reward per round } (V_{R/\text{round}})$$

$$V_{R/\text{round}} = 128 \text{ TRX/block} \times 20 \text{ blocks/min} \times 60 \text{ mins/hr} \times 6 \text{ hrs/round}$$

참고: 이는 WITNESS_127_PAY_PER_BLOCK = 128 TRX로 설정되어 있으며, 동적 네트워크 매개변수(Dynamic Network Parameters)에 의해 정의된다.

연간 총 투표 보상(Total vote reward per year)

왜 매년 1,345,536,000 TRX인가?

$$1,345,536,000 \text{ TRX} = \text{total vote reward per year } (V_{R/\text{year}})$$

$$V_{R/\text{year}} = 921,600 \text{ TRX/round} \times 4 \text{ rounds/day} \times 365 \text{ days/year}$$

b. 블록 보상(Block Reward)

블록 보상은 슈퍼 대표 보상(Super Representative Reward)이라고도 하며, 매 라운드(6시간)마다 선출된 상위 27명의 후보(SR)가 약 57,600 TRX를 채굴된 형태로 공유한다. 보상은 27명의 SR에게 균등하게 분배되며(네트워크 오류로 인해 놓친 블록 수만큼은 차감된다), 매년 총 84,096,000 TRX가 27명의 SR에게 지급된다.

라운드당 총 블록 보상(Total block reward per round)

왜 매 라운드마다 57,600 TRX인가?

$$57,600 \text{ TRX} = \text{total block reward per round } (BR/\text{round})$$

$$BR/\text{round} = 8 \text{ TRX/block} \times 20 \text{ blocks/min} \times 60 \text{ mins/hr} \times 6 \text{ hrs/round}$$

참고: 단위 블록 보상은 WITNESS_PAY_PER_BLOCK = 8 TRX로 설정되어 있다. 자세한 내용은 동적 네트워크 매개변수를 참조하라.

연간 총 블록 보상(Total block reward per year)

왜 매년 84,096,000 TRX인가?

$$84,096,000 \text{ TRX} = \text{total block reward per year } (BR/\text{year})$$

$$BR/year = 57,600 \text{ TRX/round} \times 4 \text{ rounds/day} \times 365 \text{ days/year}$$

c. 보상 계산(Reward Calculation)

SR 보상 계산(SR reward calculation)

$$total \ reward = vote \ reward(VR) + block \ reward(BR)$$

$$VR = total \ VR \times \frac{votes \ SR \ candidate \ received}{total \ votes}$$

$$BR = \frac{total \ BR}{27} - block \ missed \times 32$$

참고: 보상은 SR별로 매 라운드(6시간)마다 계산된다.

순위 28위~127위 SR 후보 보상 계산(Rank 28 to rank 127 SR candidate reward calculation)

$$total \ reward = vote \ reward(VR)$$

$$VR = total \ VR \times \frac{votes \ SR \ candidate \ received}{total \ votes}$$

참고: 보상은 SR 후보별로 매 라운드(6시간)마다 계산된다.

9.2 위원회(Committee)

9.2.1 일반 사항(General)

위원회는 블록 생성 보상, 트랜잭션 수수료 등 TRON의 동적 네트워크 매개변수를 수정하는 역할을 한다. 위원회는 현재 라운드의 27명의 슈퍼 대표(SR)로 구성된다. 각 SR은 제안을 제출하고 투표할 권리를 가진다. 하나의 제안이 19표 이상을 얻으면 승인되며, 새로운 네트워크 매개변수는 다음 유지보수 주기(3일)부터 적용된다.

9.2.2 동적 네트워크 매개변수(Dynamic Network Parameters)

자세한 내용은 부록 B를 참조하라.

9.2.3 제안 생성(Create Proposal)

동적 네트워크 매개변수의 변경을 제안할 권리는 SR 계정에만 부여된다.

9.2.4 제안 투표(Vote Proposal)

위원회 구성원(SR)만 제안에 대해 투표할 수 있으며, 기간 내에 투표하지 않은 구성원은 반대표를 던진 것으로 간주된다. 제안은 생성된 시점부터 3일 동안 활성 상태를 유지한다. 투표는 3일의 투표 기간 동안 변경하거나 철회할 수 있다. 기간이 종료되면 제안은 19표 이상을 얻을 경우 승인되며, 그렇지 않으면 실패로 종료된다.

9.2.5 제안 취소(Cancel Proposal)

제안자는 제안이 효력을 발생하기 전에 이를 취소할 수 있다.

9.3 구조(Structure)

슈퍼 대표(SR)는 새로 생성된 블록의 증인(Witness) 역할을 수행한다. 하나의 증인에는 다음 8개의 매개 변수가 포함된다.

1. **address**: 해당 증인의 주소 — 예: 0xu82h...7237.
2. **voteCount**: 해당 증인이 받은 투표 수 — 예: 234234.
3. **pubKey**: 해당 증인의 공개 키 — 예: 0xu82h...7237.
4. **url**: 해당 증인의 URL — 예: https://www.noonetrust.com.
5. **totalProduced**: 해당 증인이 생성한 블록 수 — 예: 2434.
6. **totalMissed**: 해당 증인이 놓친 블록 수 — 예: 7.
7. **latestBlockNum**: 가장 최근 블록의 높이 — 예: 4522.
8. **isJobs**: 불리언 플래그(Boolean flag).

```
1 message Witness{
2     bytes address = 1;
3     int64 voteCount = 2;
4     bytes pubKey = 3;
5     string url = 4;
6     int64 totalProduced = 5;
7     int64 totalMissed = 6;
8     int64 latestBlockNum = 7;
9     bool isJobs = 8;
10 }
```

10 DApp 개발(DApp Development)

10.1 API

TRON 네트워크는 풀 노드(Full Node)를 통해 네트워크와 상호작용할 수 있는 60개 이상의 HTTP API 게이트웨이를 제공한다. 또한 TronWeb은 개발자가 스마트 컨트랙트를 배포하고, 블록체인 상태를 변경하며, 블록체인 및 컨트랙트 정보를 조회하고, DEX에서 거래를 수행하는 등 다양한 기능을 수행할 수 있도록 하는 종합적인 자바스크립트(Javascript) 라이브러리이다. 이러한 API 게이트웨이는 로컬 프라이빗넷(Privatenet), Shasta 테스트넷, Nile 테스트넷 또는 TRON 메인넷(Mainnet)을 대상으로 설정할 수 있다.

10.2 네트워크(Networks)

TRON은 Shasta 테스트넷, Nile 테스트넷, 그리고 메인넷을 운영한다. 개발자는 노드를 배포하거나 TronGrid 서비스를 통해 풀노드 API와 상호작용하여 네트워크에 연결할 수 있다. TronGrid 서비스는 전 세계 AWS 서버에 호스팅된 부하 탈중앙화된 노드 클러스터로 구성되어 있다. DApp 개발이 확장되고 API 호출량이 증가함에 따라 TronGrid는 증가한 API 트래픽을 효율적으로 처리한다.

10.3 리소스(Resources)

TRON 개발자 허브(TRON Developer Hub)는 TRON 네트워크 위에서 개발하려는 개발자를 위해 구성된 종합 API 문서¹⁰⁸ 사이트이다. 개발자 허브는 TRON에 대한 고수준의 개념적 이해를 제공하며, 네트워크와 상호작용하는 세부 과정도 단계별로 안내한다. 문서에는 노드 설정, 스마트 컨트랙트 배포 및 상호 작용, API 구현 및 연동, 샘플 DApp 구축, 그리고 각종 개발자 도구 사용법에 대한 가이드가 포함되어 있다.

¹⁰⁸ TRON 개발자 허브: <https://developers.tron.network>

11 결론(Conclusion)

TRON은 기존 블록체인 네트워크가 직면한 문제를 해결하기 위해 혁신적인 방식을 적용한 확장 가능한 블록체인 솔루션이다. 하루 200만 건 이상의 트랜잭션, 70만 개 이상의 TRX 계정, 그리고 2000 TPS를 초과하는 성능을 달성함으로써, TRON은 커뮤니티가 탈중앙화되고 민주적인 네트워크를 구축할 수 있도록 지원하였다.

부록 A: 용어(Appendix A: Terminology)

주소/지갑(Address/Wallet)

TRON 네트워크에서 계정 자격 증명으로 구성된 주소 또는 지갑은 개인 키와 공개 키로 이루어진 키 쌍을 통해 생성된다. 공개 키는 개인 키로부터 알고리즘을 통해 파생되며, 주로 세션 키 암호화, 서명 검증, 그리고 해당 개인 키로 복호화할 수 있는 데이터 암호화에 사용된다.

애플리케이션 바이너리 인터페이스(Application Binary Interface, ABI)

ABI는 두 개의 바이너리 프로그램 모듈 간의 인터페이스를 의미한다. 일반적으로 이러한 모듈 중 하나는 라이브러리 또는 운영체제 기능이며, 다른 하나는 사용자에게 의해 실행되는 프로그램이다.

애플리케이션 프로그래밍 인터페이스(Application Programming Interface, API)

API는 주로 사용자 클라이언트 개발에 사용된다. API 지원을 통해 개발자 스스로 토큰 발행 플랫폼을 설계할 수도 있다.

자산(Asset)

TRON 문서에서 자산(Asset)은 토큰(Token)과 동일한 의미를 가지며, TRC-10 토큰으로도 표시된다.

대역폭 포인트(Bandwidth Points, BP)

네트워크가 원활하게 작동하도록 하기 위해 TRON 네트워크 트랜잭션은 연료로서 BP를 사용한다. 각 계정은 TRON 체인 매개변수 API의 `freeNetLimit` 파라미터로 정의된 일일 무료 BP를 받으며, TRX를 동결하여 추가 BP를 얻을 수 있다. TRX 및 TRC-10 토큰 전송은 모두 BP를 소비하는 일반 트랜잭션이다. 스마트 컨트랙트의 배포 및 실행 트랜잭션은 BP와 에너지(Energy)를 모두 소비한다.

블록(Block)

블록은 트랜잭션의 디지털 기록을 포함한다. 하나의 완전한 블록은 매직 넘버(Magic Number), 블록 크기(Block Size), 블록 헤더(Block Header), 트랜잭션 카운터(Transaction Counter), 트랜잭션 데이터(Transaction Data)로 구성된다.

블록 보상(Block Reward)

블록 생성 보상은 하위 계정(주소/지갑)으로 전송된다. SR은 Tronscan 또는 API를 통해 직접 보상을 청구할 수 있다.

블록 헤더(Block Header)

블록 헤더는 블록의 일부이며, TRON 블록 헤더에는 이전 블록의 해시, 머클 루트(Merkle Root), 타임스탬프, 버전, 증인 주소가 포함된다.

콜드 지갑(Cold Wallet)

콜드 지갑은 오프라인 지갑(Offline Wallet)이라고도 하며, 개인 키를 네트워크와 완전히 분리된 상태로 유지한다. 콜드 지갑은 일반적으로 인터넷에 연결되지 않은 “콜드” 장치(예: 오프라인 상태의 컴퓨터나 모바일 기기)에 설치되어 TRX 개인 키의 보안을 보장한다.

탈중앙화 애플리케이션(Decentralized Application, DApp)

DApp은 중앙에서 신뢰되는 제3자 없이 운영되는 애플리케이션이다. 이는 중개자 없이 최종 사용자 간 또는 사용자와 자원 간 직접적인 상호작용, 합의, 또는 통신을 가능하게 하는 애플리케이션이다.

gRPC 원격 프로시저 호출(gRPC Remote Procedure Calls, gRPC)

gRPC¹¹⁾는 구글(Google)에서 처음 개발된 오픈소스 원격 프로시저 호출(RPC) 시스템이다. gRPC는 HTTP/2를 전송 프로토콜로 사용하고, 인터페이스 정의 언어로 프로토콜 버퍼(Protocol Buffers)를 채택하며, 인증(Authentication), 양방향 스트리밍 및 흐름 제어(Flow Control), 동기/비동기 바인딩(Blocking/Nonblocking Bindings), 취소(Cancellation) 및 타임아웃(Timeouts) 등의 기능을 제공한다. 또한 다양한 언어에 대해 크로스 플랫폼 클라이언트 및 서버 바인딩을 자동 생성한다. 일반적인 사용 사례로는 마이크로서비스(Microservices) 아키텍처 내 서비스 간 연결, 모바일 디바이스 및 브라우저 클라이언트를 백엔드 서비스와 연결하는 경우가 있다.

핫 지갑(Hot Wallet)

핫 지갑은 온라인 지갑(Online Wallet)이라고도 하며, 사용자의 개인 키가 온라인에서 사용될 수 있도록 허용한다. 따라서 잠재적인 취약점이나 악의적 행위자에 의한 가로채기 위험에 노출될 수 있다.

자바 개발 키트(Java Development Kit, JDK)

JDK는 자바 애플리케이션을 위한 개발 도구 세트(Java SDK)로, 자바 개발의 핵심을 이루며 자바 애플리케이션 환경(JVM + 자바 클래스 라이브러리)과 자바 개발 도구를 포함한다.

KhaosDB

TRON은 풀노드 메모리에 KhaosDB를 탑재하고 있으며, 이는 일정 기간 동안 생성된 모든 새로운 포크 체인을 저장할 수 있다. 또한 증인(Witness)이 자신이 운영 중인 활성 체인에서 새로운 메인 체인으로 신속하게 전환할 수 있도록 지원한다. 자세한 내용은 2.2.2 상태 스토리지(State Storage)를 참조하라.

LevelDB

LevelDB는 빠른 읽기/쓰기(R/W) 성능과 신속한 개발 요구사항을 충족하기 위해 처음 채택되었다. 메인넷(Mainnet) 런칭 이후, TRON은 자체 요구에 맞게 완전히 맞춤화된 데이터베이스로 업그레이드하였다. 자세한 내용은 2.2.1 블록체인 스토리지(Blockchain Storage)를 참조하라.

머클 루트(Merkle Root)

머클 루트는 블록체인 네트워크 내 블록에 포함된 모든 트랜잭션 해시들의 최종 해시 값이다. 자세한 내용은 3.1 위임 지분 증명(DPoS)을 참조하라.

공개 테스트넷(Public Testnet, Shasta, Nile)

단일 노드 구성에서 실행되는 네트워크 버전이다. 개발자는 경제적 손실에 대한 우려 없이 네트워크 기능을 연결하고 테스트할 수 있다. 테스트넷 토큰은 실제 가치가 없으며, 누구나 퍼블릭 파우셋(Public Faucet)을 통해 추가로 요청할 수 있다.

원격 프로시저 호출(remote procedure call, RPC)

탈중앙화된 컴퓨팅에서 RPC¹¹는 한 컴퓨터 프로그램이 다른 주소 공간(일반적으로 공유 네트워크 내의 다른 컴퓨터)에서 프로시저(서브루틴)를 실행하도록 하는 과정으로, 프로그래머가 원격 상호작용의 세부 사항을 명시적으로 코드화하지 않아도 마치 로컬 프로시저를 호출하는 것처럼 사용할 수 있는 개념이다.

확장성(Scalability)

확장성은 TRON 프로토콜의 특징 중 하나로, 시스템, 네트워크 또는 프로세스가 증가하는 작업량을 처리하거나 그 성장을 수용할 수 있도록 확장될 수 있는 능력을 의미한다.

¹¹ <https://en.wikipedia.org/wiki/GRPC>

¹² https://en.wikipedia.org/wiki/Remote_procedure_call

SUN

SUN은 TRX의 최소 단위로, 이전의 drop 단위를 대체하였다. 1 TRX = 1,000,000 SUN이다.

처리량(Throughput)

높은 처리량은 TRON 메인넷의 특징 중 하나로, 초당 트랜잭션 수(TPS, Transactions Per Second)로 측정된다. 즉, 1초 동안 처리할 수 있는 최대 트랜잭션 용량을 의미한다.

타임스탬프(Timestamp)

블록 생성의 대략적인 시간은 유닉스 타임스탬프(Unix Timestamp)로 기록되며, 이는 1970년 1월 1일 00:00:00 UTC 이후 경과한 밀리초(ms) 단위의 시간을 나타낸다.

TRC-10

TRON 플랫폼에서 사용되는 암호화 토큰 표준이다. TRON 블록체인에서 초기 코인 공개(ICO)를 진행할 때는 일정한 규칙과 인터페이스를 따라야 한다.

TRX

TRX는 트로닉스(Tronix)의 약자로, TRON의 공식 암호화폐를 의미한다.

부록 B: 동적 네트워크 매개변수 (Appendix B: Dynamic Network Parameters)

0. MAINTENANCE_TIME_INTERVAL

- **설명(Description):** 유지보수 간격 시간을 수정하며, 이는 SR이 투표 보상을 계산하고 분배하는 간격이기도 하다.
- **예시(Example):** $[6 * 3600 * 1000]$ ms — 즉 6시간.
- **범위(Range):** $[3 * 27 * 1000, 24 * 3600 * 1000]$ ms

1. ACCOUNT_UPGRADE_COST

- **설명(Description):** 계정이 슈퍼 대표(SR) 후보로 지원할 때의 비용을 수정한다.
- **예시(Example):** $[9,999,000,000]$ SUN — 즉 9,999 TRX.
- **범위(Range):** $[0, 100,000,000,000,000,000]$ SUN

2. CREATE_ACCOUNT_FEE

- **설명(Description):** 새로운 주소로 TRX를 전송하여 새 계정을 생성할 때 발생하는 수수료를 수정한다.
- **예시(Example):** $[100,000]$ SUN — 즉 0.1 TRX.
- **범위(Range):** $[0, 100,000,000,000,000,000]$ SUN

3. TRANSACTION_FEE

- **설명(Description):** 계정의 무료/스테이킹 대역폭이 부족할 때 대역폭 포인트를 소비하는 수수료 비율을 수정한다.
- **예시(Example):** $[10]$ SUN/byte.
- **범위(Range):** $[0, 100,000,000,000]$ SUN/byte

4. ASSET_ISSUE_FEE

- **설명(Description):** 새로운 TRC-10 토큰을 발행할 때 발생하는 일회성 수수료를 수정한다.
- **예시(Example):** $[1,024,000,000]$ SUN — 즉 1,024 TRX.
- **범위(Range):** $[0, 100,000,000,000,000,000]$ SUN

5. WITNESS_PAY_PER_BLOCK

- **설명(Description):** 블록을 성공적으로 생성한 슈퍼 대표(SR)에게 지급되는 보상액을 수정한다.
- **예시(Example):** $[16,000,000]$ SUN — 즉 16 TRX.
- **범위(Range):** $[0, 100,000,000,000,000,000]$ SUN

6. WITNESS_STANDBY_ALLOWANCE

- **설명(Description):** 매 라운드마다 상위 127명의 슈퍼 대표 및 파트너에게 분배되는 총 보상액을 수정한다.
- **예시(Example):** $[115,200,000,000]$ SUN — 즉 115,200 TRX.
- **범위(Range):** $[0, 100,000,000,000,000,000]$ SUN

7. CREATE_NEW_ACCOUNT_FEE_IN_SYSTEM_CONTRACT

- **설명(Description):** 시스템 컨트랙트 호출을 통해 새 계정을 생성할 때 부과되는 수수료를 수정한다.
- **예시(Example):** [0] SUN.
- **범위(Range):** [0, 100,000,000,000,000,000] SUN

8. CREATE_NEW_ACCOUNT_BANDWIDTH_RATE

- **설명(Description):** 새 계정을 생성할 때 소비되는 대역폭(Bandwidth) 배율을 수정한다.
- **예시(Example):** [1].
- **범위(Range):** [0, 100,000,000,000,000,000]

9. ALLOW_CREATION_OF_CONTRACTS

- **설명(Description):** 새로운 스마트 컨트랙트 생성 기능을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

10. REMOVE_THE_POWER_OF_THE_GR

- **설명(Description):** 제네시스 대표(Genesis Representatives, GR)가 보유한 초기 투표 권한을 제거한다.
- **예시(Example):** True
- **유형(Type):** True/False — **참고:** True로 설정된 이후에는 False로 되돌릴 수 없다.

11. ENERGY_FEE

- **설명(Description):** 계정의 에너지(Energy)가 부족할 때 에너지를 소비하는 수수료 비율을 수정한다.
- **예시(Example):** [10] SUN.
- **범위(Range):** [0, 100,000,000,000,000,000] SUN

12. EXCHANGE_CREATE_FEE

- **설명(Description):** 탈중앙화 거래소(DEX)에서 새로운 TRC-10 거래 쌍을 생성할 때의 수수료를 수정한다.
- **예시(Example):** [1,024,000,000] SUN — 즉 1,024 TRX.
- **범위(Range):** [0, 100,000,000,000,000,000] SUN

13. MAX_CPU_TIME_OF_ONE_TX

- **설명(Description):** 단일 트랜잭션에 허용되는 최대 CPU 실행 시간을 수정한다.
- **예시(Example):** [50] ms.
- **범위(Range):** [0, 1000] ms

14. ALLOW_UPDATE_ACCOUNT_NAME

- **설명(Description):** 계정이 이름을 변경할 수 있는 기능을 활성화하거나 비활성화한다.
- **예시(Example):** False
- **유형(Type):** True/False

15. ALLOW_SAME_TOKEN_NAME

- **설명(Description):** 동일한 이름을 가진 서로 다른 TRC-10 토큰의 생성을 허용한다.
- **예시(Example):** True
- **유형(Type):** True/False

16. ALLOW_DELEGATE_RESOURCE

- **설명(Description):** 자원 위임 기능(Resource Delegation, 스테이킹 v1.0)을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

18. ALLOW_TVM_TRANSFER_TRC10

- **설명(Description):** 스마트 컨트랙트가 `transferToken` 호출을 사용하여 TRC-10 토큰을 네이티브 방식으로 전송할 수 있도록 허용한다.
- **예시(Example):** True
- **유형(Type):** True/False

19. TOTAL_CURRENT_ENERGY_LIMIT

- **설명(Description):** 네트워크에서 현재 사용 가능한 총 에너지(Energy) 양을 수정한다.
- **예시(Example):** [50,000,000,000].
- **범위(Range):** [0, 100,000,000,000,000,000]

20. ALLOW_MULTI_SIGN

- **설명(Description):** 계정의 다중 서명(Multi-Signature) 기능을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

21. ALLOW_ADAPTIVE_ENERGY

- **설명(Description):** 적응형 에너지 모델(Adaptive Energy Model)을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

22. UPDATE_ACCOUNT_PERMISSION_FEE

- **설명(Description):** 계정의 권한 구조를 업데이트할 때 부과되는 수수료를 수정한다.
- **예시(Example):** [100,000,000] SUN — 즉 100 TRX.
- **범위(Range):** [0, 100,000,000,000] SUN

23. MULTI_SIGN_FEE

- **설명(Description):** 다중 서명(Multi-Signature) 트랜잭션을 처리할 때 부과되는 추가 수수료를 수정한다.
- **예시(Example):** [1,000,000] SUN — 즉 1 TRX.
- **범위(Range):** [0, 100,000,000,000] SUN

24. ALLOW_PROTO_FILTER_NUM

- **설명(Description):** 프로토콜 메시지 필터링 기능을 활성화하거나 비활성화한다.
- **예시(Example):** False
- **유형(Type):** True/False

26. ALLOW_TVM_CONSTANTINOPLE

- **설명(Description):** TVM에서 이더리움 콘스탄티노플(Constantinople) 업그레이드의 기능을 활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

29. ADAPTIVE_RESOURCE_LIMIT_MULTIPLIER

- **설명(Description):** 적응형 자원 모델(Adaptive Resource Model)에서 사용되는 배수(Multiplier)를 수정한다.
- **예시(Example):** [1000].
- **범위(Range):** [1, 10000]

30. ALLOW_CHANGE_DELEGATION

- **설명(Description):** 사용자가 언스테이킹(Unstaking) 없이 위임된 자원의 수신자를 변경할 수 있도록 허용한다.
- **예시(Example):** True
- **유형(Type):** True/False

31. WITNESS_127_PAY_PER_BLOCK

- **설명(Description):** 상위 28~127위 파트너에게 지급되는 블록 생성 보상을 수정한다.
- **예시(Example):** [160,000,000] SUN — 즉 160 TRX.
- **범위(Range):** [0, 100,000,000,000,000,000] SUN

32. ALLOW_TVM_SOLIDITY_059

- **설명(Description):** Solidity v0.5.9 이상 버전으로 컴파일된 스마트 컨트랙트를 지원하도록 TVM 기능을 활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

33. ADAPTIVE_RESOURCE_LIMIT_TARGET_RATIO

- **설명(Description):** 적응형 자원 모델(Adaptive Resource Model) 알고리즘에서 사용되는 목표 비율(Target Ratio)을 수정한다.
- **예시(Example):** [10].
- **범위(Range):** [1, 1000]

35. FORBID_TRANSFER_TO_CONTRACT

- **설명(Description):** 페이저블 폴백 함수(Payable Fallback Function)가 없는 컨트랙트로의 직접적인 TRX 전송을 금지한다.
- **예시(Example):** True
- **유형(Type):** True/False

39. ALLOW_SHIELDED_TRC20_TRANSACTION

- **설명(Description):** TRC-20 토큰에 대한 프라이버시 보호형 실드 트랜잭션(Shielded Transaction)을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

40. ALLOW_PBFT

- **설명(Description):** 더 빠른 트랜잭션 파이널리티를 위해 PBFT 합의 메커니즘을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

41. ALLOW_TVM_ISTANBUL

- **설명(Description):** TVM에서 이더리움 이스탄불(Istanbul) 업그레이드 기능을 활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

44. ALLOW_MARKET_TRANSACTION

- **설명(Description):** 탈중앙화 거래소(DEX)에서 시장가 주문 트랜잭션(Market Order Transaction)을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

45. MARKET_SELL_FEE

- **설명(Description):** 탈중앙화 거래소(DEX)에서 매도 주문을 실행할 때 부과되는 수수료를 수정한다.
- **예시(Example):** [0] SUN.
- **범위(Range):** [0, 10,000,000,000] SUN

46. MARKET_CANCEL_FEE

- **설명(Description):** 탈중앙화 거래소(DEX)에서 열린 주문을 취소할 때 발생하는 수수료를 수정한다.
- **예시(Example):** [0] SUN.
- **범위(Range):** [0, 10,000,000,000] SUN

47. MAX_FEE_LIMIT

- **설명(Description):** 사용자가 단일 트랜잭션에 대해 설정할 수 있는 최대 `fee_limit` 값을 수정한다.
- **예시(Example):** [15,000,000,000] SUN — 즉 15,000 TRX.
- **범위(Range):** [0, 100,000,000,000,000,000] SUN

48. ALLOW_TRANSACTION_FEE_POOL

- **설명(Description):** 트랜잭션 수수료 풀(Transaction Fee Pool) 메커니즘을 활성화하거나 비활성화한다.
- **예시(Example):** False
- **유형(Type):** True/False

49. ALLOW_BLACKHOLE_OPTIMIZATION

- **설명(Description):** 토큰 소각 시 에너지 소비를 최소화하는 최적화 기능을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

51. ALLOW_NEW_RESOURCE_MODEL

- **설명(Description):** 새로운 자원 모델(New Resource Model, 스테이킹 2.0)을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

52. ALLOW_TVM_FREEZE

- **설명(Description):** 새로운 스테이킹/자원 모델(스테이킹 2.0)과 관련된 TVM 명령어를 활성화하거나 비활성화한다.

- 예시(Example): True
- 유형(Type): True/False

53. ALLOW_ACCOUNT_ASSET_OPTIMIZATION

- 설명(Description): 계정 자산(TRC-10) 잔액의 저장 방식을 최적화하는 기능을 활성화한다.
- 예시(Example): True
- 유형(Type): True/False

59. ALLOW_TVM_VOTE

- 설명(Description): 스마트 컨트랙트가 투표 연산을 실행할 수 있도록 허용한다.
- 예시(Example): True
- 유형(Type): True/False

60. ALLOW_TVM_COMPATIBLE_EVM

- 설명(Description): TVM의 EVM 호환 기능을 활성화하거나 비활성화한다.
- 예시(Example): True
- 유형(Type): True/False

61. FREE_NET_LIMIT

- 설명(Description): 각 계정이 매일 받는 무료 대역폭 포인트(Bandwidth Points)의 양을 수정한다.
- 예시(Example): [5000].
- 범위(Range): [0, 100000]

62. TOTAL_NET_LIMIT

- 설명(Description): 스테이킹된 TRX에서 네트워크가 제공하는 총 대역폭(Bandwidth)의 양을 수정한다.
- 예시(Example): [43,200,000,000].
- 범위(Range): [0, 1,000,000,000,000]

63. ALLOW_TVM_LONDON

- 설명(Description): TVM에서 이더리움 런던(London) 업그레이드 기능을 활성화한다.
- 예시(Example): True
- 유형(Type): True/False

65. ALLOW_HIGHER_LIMIT_FOR_MAX_CPU_TIME_OF_ONE_TX

- 설명(Description): MAX_CPU_TIME_OF_ONE_TX 매개변수에 대해 더 높은 최대값을 제안할 수 있도록 허용한다.
- 예시(Example): True

- **유형(Type):** True/False

66. ALLOW_ASSET_OPTIMIZATION

- **설명(Description):** 계정 자산 저장 최적화 기능을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

67. ALLOW_NEW_REWARD

- **설명(Description):** 슈퍼 대표(SR) 및 투표자에 대한 새로운 보상 알고리즘을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

68. MEMO_FEE

- **설명(Description):** 트랜잭션에 메모(Memo)를 포함할 때 바이트당 부과되는 수수료를 수정한다.
- **예시(Example):** [1,000,000] SUN — 즉 1 TRX.
- **범위(Range):** [0, 1,000,000,000]

69. ALLOW_DELEGATE_OPTIMIZATION

- **설명(Description):** 자원 위임 저장소(Resource Delegation Storage)에 대한 최적화 기능을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

70. UNFREEZE_DELAY_DAYS

- **설명(Description):** 새로운 스테이킹 메커니즘 사용 시 자산이 잠금 상태로 유지되는 일 수를 수정한다.
- **예시(Example):** [14]일.
- **범위(Range):** [1, 365]일

71. ALLOW_OPTIMIZED_RETURN_VALUE_OF_CHAIN_ID

- **설명(Description):** CHAINID 오퍼코드(opcode)의 반환값을 최적화하는 기능을 활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

72. ALLOW_DYNAMIC_ENERGY

- **설명(Description):** 동적 에너지(Dynamic Energy) 모델을 활성화하거나 비활성화한다.
- **예시(Example):** True

- 유형(Type): True/False

73. DYNAMIC_ENERGY_THRESHOLD

- 설명(Description): 동적 페널티가 적용되기 전의 컨트랙트 에너지 소비 임계값을 수정한다.
- 예시(Example): [5,000,000,000].
- 범위(Range): [0, 9223372036854775807]

74. DYNAMIC_ENERGY_INCREASE_FACTOR

- 설명(Description): 임계값을 초과한 컨트랙트의 에너지 비용 증가 비율(%)을 수정한다.
- 예시(Example): [2000].
- 범위(Range): [0, 10000]

75. DYNAMIC_ENERGY_MAX_FACTOR

- 설명(Description): 동적 에너지 모델에서 에너지 비용의 최대 배수를 수정한다.
- 예시(Example): [34000].
- 범위(Range): [0, 100000]

76. ALLOW_TVM_SHANGHAI

- 설명(Description): TVM에서 이더리움 상하이(Shanghai) 업그레이드 기능을 활성화한다.
- 예시(Example): True
- 유형(Type): True/False

77. ALLOW_CANCEL_ALL_UNFREEZE_V2

- 설명(Description): 스테이킹 2.0(Staking 2.0) 환경에서 보류 중인 모든 언스테이킹(Unstaking) 요청을 취소할 수 있는 기능을 활성화한다.
- 예시(Example): True
- 유형(Type): True/False

78. MAX_DELEGATE_LOCK_PERIOD

- 설명(Description): 자원 위임(Resource Delegation)의 최대 잠금 기간을 수정한다.
- 예시(Example): [864000].
- 범위(Range): [86400, 10512000]

79. ALLOW_OLD_REWARD_OPT

- 설명(Description): 기존 보상 시스템(Old Reward System)의 보상 인출 알고리즘에 대한 최적화 기능을 활성화한다.
- 예시(Example): True
- 유형(Type): True/False

81. ALLOW_ENERGY_ADJUSTMENT

- **설명(Description):** 에너지 조정(Energy Adjustment) 기능을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

82. MAX_CREATE_ACCOUNT_TX_SIZE

- **설명(Description):** 새 계정을 생성하는 트랜잭션의 최대 크기를 수정한다.
- **예시(Example):** [1000].
- **범위(Range):** [500, 10000]

83. ALLOW_TVM_CANCUN

- **설명(Description):** TVM에서 이더리움 칸쿤(Cancun) 업그레이드 기능을 활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

87. ALLOW_STRICT_MATH

- **Description:** Migrate TVM math operations to `java.lang.StrictMath` for cross-platform consistency.
- **Example:** True
- **Type:** True/False

88. CONSENSUS_LOGIC_OPTIMIZATION

- **Description:** Enable or disable optimizations in the consensus logic.
- **Example:** True
- **Type:** True/False

89. ALLOW_TVM_BLOB

- **Description:** Enable or disable TVM support for blob transactions (EIP-4844).
- **Example:** True
- **Type:** True/False

87. ALLOW_STRICT_MATH

- **설명(Description):** 교차 플랫폼 일관성을 위해 TVM의 수학 연산을 `java.lang.StrictMath`로 마이그레이션한다.
- **예시(Example):** True
- **유형(Type):** True/False

88. CONSENSUS_LOGIC_OPTIMIZATION

- **설명(Description):** 합의 로직(Consensus Logic)의 최적화 기능을 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False

89. ALLOW_TVM_BLOB

- **설명(Description):** TVM에서 블롭 트랜잭션(Blob Transaction, EIP-4844)을 지원하도록 활성화하거나 비활성화한다.
- **예시(Example):** True
- **유형(Type):** True/False