

토카막 네트워크

튜링 완전하고 확장성 높은
플라즈마 블록체인 빌딩 프로토콜

Onther Inc.

May, 2019

Contents

초록	4
면책 조항	5
1 서론	8
2 관련 연구	9
2.1 플라즈마(Plasma)	9
2.1.1 단순 전송(Simple Transfer)	9
2.1.1.1 플라즈마 MVP	9
2.1.1.2 플라즈마 Cash	9
2.1.1.3 플라즈마 Group	10
2.1.2 일반 상태 변화(General State Transition)	10
2.1.2.1 플라즈마 Leap	10
2.1.2.2 플라즈마 EVM	10
2.1.3 플라즈마 상태 검증(State Verification)	10
2.1.3.1 트루빗 프로토콜(TrueBit Protocol)	10
2.2 퍼블릭 블록체인의 트랜잭션 수수료 모델	11
2.2.1 이더리움(Ethereum)	11
2.2.2 스팀(Steem)	11
2.2.3 이오스(EOS)	11
3 시스템 구성	13
3.1 일반 상태 변화 강제(General State Transition Enforcement)	13
3.1.1 일반 상태의 개념과 연산 강제	13
3.1.2 요청 가능 컨트랙트 : 진입과 탈출 규칙의 정의(Requestable : Defining Entering and Exiting Rules)	14
3.1.2.1 숫자 카운터(Number Counter)	14
3.1.2.1.1 단순 카운터(Simple Counter)	15
3.1.2.1.2 동결 되는 카운터(FreezableCounter)	16
3.1.2.1.3 추적 가능한 카운터(Trackable Counter)	17
3.1.2.2 토큰(Token)	17

3.1.2.2.1	요청 가능한 단순 토큰(RequestableSimpleToken)	18
3.1.2.2.2	요청가능한 ERC20 래퍼(RequestableERC20Wrapper)	18
3.1.3	검증게임 : 챌린지(Verification Game : Challenge)	20
3.1.3.1	검증자와 검증(Verifier and Verify)	20
3.1.3.2	탈출 챌린지(Exit Challenge)	21
3.1.3.3	프로토콜 챌린지 : 널-계정, 탈출, 최소가스가격(Protocol Challenge : Null-Address, Exit and MGP)	21
3.2	데이터 가용성 문제(Data Availability Problem)	21
3.3	무합의성(No Consensus)	22
4	경제 모델(Economic Model)	23
4.1	토큰과 주조차익(Token and Seigniorage)	23
4.1.1	토큰의 사용과 단위	23
4.1.2	토큰 주조차익(Token Seigniorage)	23
4.1.3	주조차익 분배(Seigniorage Distribution)	24
4.1.4	이더(Ether)와 토파막 네트워크 토큰	24
4.2	플라즈마 체인 예치금(Chain Staking)	24
4.3	플라즈마 트랜잭션 수수료	25
4.3.1	시빌 공격과 트랜잭션 수수료(Sybil Attack and Transaction Fee)	25
4.3.2	스태미나 : 수수료 위임과 재생	26
4.3.3	최소 가스 가격(Minimum Gas Price; MGP)	26
5	탈중앙화 어플리케이션 블록체인	28
5.1	플라즈마화된 탈중앙화 어플리케이션(Plasma DApps)	28
5.1.1	탈중앙화된 거래소(Decentralized exchange; DEX)	28
5.1.1.1	암호화폐 담보대출 시스템(Compound Protocol)	29
5.1.2	크립토 키티(ERC721)	30
5.1.3	암호화폐 지갑과 결제	30
5.1.4	탈중앙화된 가치 안정화 토큰(MakerDAO)	31
5.1.4.1	Native Stable Coin	32
5.1.4.2	Non-Native Stable Coin	32
5.2	도메인 특정 플라즈마	33
5.2.1	프라이버시	33
5.2.1.1	익명화된 토큰(Anonymous ERC20)	33
5.2.1.2	Quorum 기반 플라즈마	34
5.2.2	클라우드 저장소	34
5.3	파생 비즈니스	35
5.3.1	수수료 대출 마켓	35
5.3.2	빈 블록을 제출하는 플라즈마 오퍼레이터(플라즈마 채굴 풀)	36

CONTENTS	3
6 기타 이슈	37
6.1 이더리움 2.0과 토카막 네트워크	37
6.1.1 작업증명 vs 지분증명(Proof of Work vs Proof of Stake)	37
6.1.2 eWasm	38
6.1.2.1 토카막 플라즈마 TPS(Transaction Per Second) 개선	38
6.1.2.2 표준 라이브러리(Standard Library)	39
6.1.2.3 토카막 플라즈마 체인에 eWasm 적용	39
6.2 플라즈마 블록체인의 확장성	39
6.3 플라즈마 체인의 다양성	40
7 결론	42
부록	43
.1 용어	43
.1.1 공통	43
.1.2 시스템 구성	43
.1.3 경제모델 및 기타	44
References	46

초록

비탈릭 부테린(Vitalik Buterin)에 의해 2015년 만들어진 이더리움(Ethereum)은 블록체인을 하나의 상태 변화 시스템(State Transition System)[6]으로 바라보았고, 스마트 컨트랙트(Smart Contract)를 이용해 블록체인 기술의 응용 분야를 확장시켰다. 하지만 이더리움 네트워크가 확장되어가는 과정에서 단위 시간당 처리할 수 있는 트랜잭션 처리량 한계에 부딪혔고, 늘어나는 데이터 용량 부담도 점차 증가하고 있다. 이러한 문제는 샤팅, 스테이트 채널, 플라즈마 등 다양한 확장성(Scalability) 솔루션들의 등장을 가속시켰다. 이더리움 블록체인은 여러번의 하드포크를 통해서 이러한 확장성 솔루션을 지원하기 위한 새로운 기능들을 도입해왔으나, 네트워크 규모가 커져가며 이해관계자가 많아진 메인체인에서는 실험적 기능을 도입하는 것이 점차 어려워지고 있다[8].

2017년 조셉 푸ن(Joseph Poon)과 비탈릭 부테린은 스마트 컨트랙트의 올바른 실행을 강제하면서도 초당 수백만건의 상태 변화를 처리할 수 있는 프레임워크로 플라즈마(Plasma)[32]를 제안했다. 이에 기초하여 다양한 플라즈마 구현체가 제안되었고, 이러한 제안은 적어도 “토큰(ERC20, ERC721)의 상태 변화 시스템”이라는 블록체인의 일부 응용분야 내에서 의미있는 성과를 만들어냈다. 하지만 마치 초기의 블록체인이 그랬듯, 대부분의 플라즈마 구현들이 “제한된 기능의 상태 변화 시스템”으로만 활용되고 있다.

토카막 네트워크가 제공하려는 것은 다양한 튜링 완전(Turing-Complete)한 플라즈마 블록체인을 손쉽게 만들어 낼 수 있는 프로토콜이다. 토카막 네트워크는 “임의의 상태”를 다르게 변화시키기 위한 규칙을 루트체인(Root chain)에 별도로 정의하여 플라즈마 체인의 “올바른 상태 변화” 기준을 마련하고자 한다. 이로써 확장된 상태 변화 시스템(레이어2)으로 플라즈마 블록체인의 활용 범위를 넓힐 수 있을 것이다. 따라서 토카막 네트워크는 이더리움 탈중앙화 어플리케이션(Decentralized Application; DApp)이 겪고 있는 확장성 문제를 해결할 수 있을 뿐만 아니라, 이더리움의 성능과 기능적 제약으로 인해 구현되지 못한 어플리케이션을 손쉽게 만들 수 있는 환경을 제공할 것으로 기대할 수 있다.

면책 조항

본 백서 및 이와 관련해 배포된 기타 문서들은 토카막 네트워크의 개발 및 응용에 활용되며, 이들 자료는 정보 제공만을 목적으로 하고, 향후 그 내용이 변경될 수 있다.

백서는 향후 진행될 프로젝트에 관한 것이다

백서는 프로젝트의 지속 지원을 위해, 백서를 작성한 온더(Onther Inc.)의 믿음을 바탕으로 한 미래 예측이 포함되어 있다. 백서에 언급된 토카막 네트워크는 현재 개발이 진행중이며, 주요 거버넌스 및 기술적 기능을 비롯해 다양한 내용들이 지속적으로 업데이트되고 있다. 톤(TON) 토큰은 백서에 명시된 목표 달성과 성과로 이어지지 않을 수도 있는 실험적 플랫폼(소프트웨어)과 관련 기술의 개발 및 활용에 기반한다. 토카막 네트워크가 완성될 경우, 백서에 명시된 내용과는 상당히 차이가 있을 수 있고, 온더(Onther Inc.)는 모든 계획, 향후 예상 또는 전망의 달성과 관련해 어떠한 보증도 하지 않으며, 본 문서의 어떠한 내용도 미래에 대한 약속으로 간주되어서는 안된다.

적격성

백서의 내용은 향후 특정 구매자들에게 직접 제공되며, 이들 구매자 외에는 누구도 백서의 수령이나 열람 대상이 아니다. 단순히 백서를 수령한 것으로는 적격성이 보장되지 않으며 프로젝트의 참여를 보장하지 않는다.

규제 대상 품목

토카막 네트워크 플랫폼, 톤(TON) 토큰이나 해당 플랫폼상에서 운영되는 모든 토큰은 유가증권이나 어떠한 법정관할 지역에서 규제 대상 품목으로 지정된 것을 의미하지 않는다. 백서는 유가증권 또는 기타 규제 대상 제품의 제공이나 권리, 투자 목적의 홍보, 초대 또는 권리증명을 위한 것이 아니다. 백서는 금융 서비스 제공 목적의 문서나 그 어떤 투자설명서, 증권신고서 등도 아니다. 톤(TON) 토큰은 플랫폼이나 소프트웨어 또는 기업의 자본, 수익, 소득 또는 수입에 대한 지분, 출자 지분, 유닛, 로열티 및 권리, 또는 모든 관할구역의 플랫폼이나 기타 공공 또는 사설 회사, 조직 및 기타 독립체와 관련된 지식재산을 표시하는 것이 아니다.

그 어떤 투자설명서, 증권신고서 등도 아니다.

백서는 톤(TON) 토큰 구매 권고문, 기타 청약 내지 청약의 권리증명을 위한 문서가 아니다. 백서는 계약이나 청약 또는 청약의 권리증명, 투자권유, 구매 결정과 관련된 것으로 의도 또는 해석되거나 간주되어서는 안되며, 백서의 제시 또는 백서 자체가 계약 및 투자 결정의 근거가 되는 것은 아니다.

리스크

톤(TON) 토큰의 구매 및 판매, 토파막 네트워크 프로젝트 참여에는 상당한 리스크가 존재한다. 구매자는 톤(TON) 토큰의 구매 및 판매 또는 토파막 네트워크 프로젝트에 참여하기에 앞서 모든 리스크에 대한 신중한 평가 및 고려를 하여야 하며, 백서는 사전 동의나 고지없이 변경될 수 있으며 그 내용이 보증되지 아니하므로, 구매자는 의사결정 전에 필요한 사항을 직접 확인하여야하고, 구매자의 의사결정으로 인한 손해에 대해서 온더(Onther Inc.)는 책임을 지지 않는다.

참여자의 책임

참여자는 스스로에게 적용되는 법령 및 하위규정, 규제, 계약에 따라 톤(TON) 토큰의 구매 및 판매, 토파막 네트워크 프로젝트의 참여가 가능한지 여부를 직접 판단하여야 하며, 온더(Onther Inc.)는 이에 대해 어떠한 보증이나 보장도 하지 아니한다. 참여자는 톤(TON) 토큰의 구매 및 판매, 토파막 네트워크 프로젝트의 참여에 요구되는 모든 종류의 인허가, 신고 등을 자신의 책임 하에 완료하여야 하며 온더(Onther Inc.)는 이와 관련하여 어떠한 책임도 부담하지 않는다. 참여자는 톤(TON) 토큰 및 토파막 네트워크 프로젝트의 참여를 통해 자금 세탁, 불법적인 통화 거래 및 기타 국제 협약 및 적용되는 법령에 따라 제한되는 활동에 어떠한 형태로든 참여하지 않는다는 것에 동의하여야 하며, 각 참여자는 톤(TON) 토큰 및 토파막 네트워크 프로젝트를 자금세탁, 테러자금 조달, 대량살상무기 확산 등과 관련한 목적으로 직·간접적으로 이용하거나 구매·판매·교환 및 처분 할 수 없다는 사실을 숙지하여야 한다.

구매자의 책임

구매자는 스스로에게 적용되는 법령 및 하위규정, 규제, 계약에 따라 톤(TON) 토큰의 구매 및 판매, 토파막 네트워크 프로젝트의 참여가 가능한지 여부를 직접 판단하여야 하며, 온더(Onther Inc.)는 이에 대해 어떠한 보증이나 보장도 하지 않는다. 구매자는 톤(TON) 토큰의 구매 및 판매, 토파막 네트워크 프로젝트의 참여에 요구되는 모든 종류의 인허가, 신고 등을 자신의 책임 하에 완료하여야 하며 온더(Onther Inc.)는 이와 관련하여 어떠한 책임도 부담하지 않는다.

백서의 관점

백서는 온더(Onther Inc.)의 관점과 견해를 담고 있으며, 이는 어떠한 법정관할 지역의 정부나 준정부, 당국 또는 규제기관 등의 공공기관의 정책이나 입장을 반영하지 않는다. 백서에 포함된 정보들의 경우 신뢰할 수 있는 출처를 통해 확보된 내용을 바탕으로 한 것이기는 하나, 온더(Onther Inc.)는 그 정확성 또는 완전함에 대해서는 보증하지 않는다.

백서의 공식 언어는 영어

백서 및 관련 자료들은 영어로만 발행된다. 번역본은 참조용일 뿐이며, 토파막 네트워크나 기타 개인의 인증을 받은 것이 아니다. 번역본의 정확성 및 완전함에 대해서는 온더(Onther Inc.)는 그 어떤 보증도 하지 않는다. 백서의 번역본과 영어 버전의 내용상 불일치가 존재하는 경우에는 영어 버전이 우선하는 것으로 한다.

제삼자 제휴와 보증을 의미하지 않는다

백서에서의 특정 기업 및 플랫폼에 대한 언급은 단순히 예시적인 차원에서 이루어진 것이다. 기업이나 플랫폼의 명칭 및 등록상표의 사용이 해당 당사자의 제휴나 보증을 뜻하는 것은 아니다.

전문가의 조언

톤(TON) 토큰 구매 또는 토카막 네트워크 프로젝트 참여 여부를 결정하기에 앞서, 필요에 따라 변호사, 회계사, 세무사 또는 기타 전문가들로부터의 상담이 권고된다.

백서와 관련해 관할 구역의 규제 당국이 검토한 것은 아니다

백서의 특정 기업, 네트워크 또는 잠재적 활용 사례들에 대한 언급은 오로지 예시적 차원에서 이루어진 것이다. 온더(Onther Inc.) 등 명확하게 언급된 파트너나 제공자를 제외하고, 기타 기업 또는 플랫폼 명칭 및 등록상표의 사용이 해당 당사자의 제휴나 보증을 뜻하는 것 또한 아니다.

1장 서론

ERC721¹를 사용한 최초의 암호화 수집품인 크립토 키티(Crypto Kitties)의 성공은 사람들에게 토큰 이외의 다양한 비금융 분야에 스마트 컨트랙트가 활용될 수 있는 가능성을 제시했다. 하지만 고양이(Kitties)의 소유권 이전, 경매, 교미 간 발생한 트랜잭션 과부하로 인해 전체 이더리움 네트워크가 포화되는 현상[15]을 겪으며 퍼블릭 블록체인의 한정된 처리용량에 대한 의구심도 동시에 안겨주었다.

또한 이더리움 개발자들은 일반 유저들의 탈중앙화 어플리케이션 접근성과 사용성을 높이기 위해 메타마스크, 마이이더월렛, 마이크립토와 같은 브라우저 익스텐션(Extension)과 모바일 기반의 맵 브라우저(DApp Browser), 맵 메신저(DApp Messenger) 등을 개발해 왔다. 하지만 UX측면에서 사용자는 매 트랜잭션 수수료를 이더(Ether)로만 직접 부담(지불)해야만하고, 이는 탈중앙화 어플리케이션의 사용성(Usability)을 현저히 떨어뜨리는 요인[21]이 된다.

토카막 네트워크는 다음의 방법을 통해 앞에서 언급된 문제들을 해결하고자 한다. 첫번째로 이더리움의 가상 머신이 적은 수의 노드로 동작하는 플라즈마 체인을 통해 동작하기 때문에 높은 성능을 낼 수 있다. 두번째는 플라즈마 체인의 적은 노드로 인해 발생하는 중앙화(Centralization) 문제를 적절한 체인 예치금(Chain Staking)과 플라즈마 컨트랙트(Plasma Contract)를 이용해 탈중앙화(Decentralize) 할 수 있다. 세번째로는 트랜잭션 수수료(Transaction fee)의 부담에 관한 문제를 대안적인 이더리움 가상 머신(Ethereum Virtual Machine; EVM)에서 처리하도록하여 유저 친화적이고 접근성 높은 어플리케이션 개발을 위한 토양을 마련할 수 있다.

토카막 네트워크는 지금까지 나온 퍼블릭 블록체인의 장점으로 꼽힌 탈중앙성, 정보의 영속성, 신뢰성을 놓치지 않으면서도 퍼블릭 체인의 약점으로 지적된 낮은 성능과 사용성 문제를 해결함과 동시에 서비스 거부공격(DoS attack)에 대한 구조적인 면역력까지 제공하게 된다. 또한 퍼블릭 체인에서 사용되는 이더리움 가상머신(EVM)을 큰 변형없이 그대로 사용하여 이미 구현되어 만들어진 수많은 맵(DApp)들의 스마트 컨트랙트를 곧바로 포팅해 구동시킬 수 있다.

¹대체 불가능한 토큰(Non-Fungible Token), <https://github.com/ethereum/EIPs/blob/master/EIPS/eip-721.md>

2장 관련 연구

2.1 플라즈마(Plasma)

플라즈마는 이더리움의 확장성 문제 해결을 위한 layer-2 solution이다. 플라즈마는 블록체인의 요약본을 주기적으로 이더리움 메인 체인에 제출하고, 이렇게 제출된 요약 증거를 바탕으로 플라즈마 블록에 문제가 생겼을 때 검증과정을 통해 이를 바로잡을 수 있다[32]. 비탈릭이 직접 제안한 MVP, Cash 등의 초기 플라즈마 연구는 토큰 등의 단순전송과 이에 대한 검증에 초점이 맞춰져 있었으나 이후 플라즈마 Leap, EVM 등의 일반 상태를 다루는 방향으로 발전 및 개선되고 있다.

2.1.1 단순 전송(Simple Transfer)

2.1.1.1 플라즈마 MVP

플라즈마 백서 공개 이후 제시된 첫번째 플라즈마 모델이다. UTXO기반의 이진 머클트리(Binary Merkle Tree) 구조를 사용하였다. 유효하지 않은 상태를 기반으로 한 탈출(Exit)을 방지하기 위해 챌린지 시스템을 두었고, 데이터 가용성(Data Availability) 문제를 해결하기 위해 확인(Confirmation)과 탈출 우선권(Exit Priority)을 두었다[9]. 하지만 트랜잭션 전송시 서명단계를 두 번이나 거쳐야 하는 확인(Confirmation) 절차로 인해 UX가 매우 좋지 않아 실용적으로 쓰이기는 어렵다. MVP는 대량 탈출(Mass Exit)과 같이 여러가지 향후 연구 과제들에 방향을 제시하는 좋은 시발점 역할을 하였다.

2.1.1.2 플라즈마 Cash

플라즈마 Cash는 코인 각각에 고유한 ID와 액면가(Denomination)를 부여하고, Sparse Merkle Tree(SMT)[16]를 활용한다. 또한 각 데이터 단말 노드의 index를 토큰 ID로 두고, 해당 토큰이 사용된 경우 그에 관련된 트랜잭션 정보를 값(Value)으로 기입하는 형태로 설계되었다[10]. 만약 해당 블록에서 사용되지 않은 토큰이라면 해당 토큰에 대한 트랜잭션 정보 또한 없으므로 값(Value)은 비어있게(Null) 된다. 이렇게 해당 토큰이 특정 블록에서 사용되었는지, 혹은 사용되지 않았는지를 SMT의 포함증명과 비포함증명 기능을 통해 확인할 수 있기 때문에 해당 토큰의 사용기록에 대해 명확하게 검증할 수 있게 되었다.

하지만 부분 지불(Partial Spending)이 불가능하다는 것과 각 코인에 대한 사용기록을 검증하는 과정이 매우 무거운 작업이 될 수 있다는 단점[31] 등이 향후 개선되어야 할 부분으로 지적되었다.

2.1.1.3 플라즈마 Group

플라즈마 Group은 플라즈마 Cash 기반의 플라즈마 체인 배포 프레임워크로, 누구나 쉽게 플라즈마 체인을 배포할 수 있도록 설계되었다. 또한 기존 플라즈마 Cash의 문제점들을 해결하기 위해 여러가지 개선안들을 도입했다. 먼저 Fixed Denomination 문제¹를 해결하기 위해 하나의 트랜잭션으로 하나의 코인이 아닌 다수의 코인(Large Range of Coins)을 전송할 수 있도록 하였다. 또한 이러한 과정에서 검증과정이 무거워지는 것을 피하기 위해 Merkle Sum Tree²를 사용하였다. 이를 통해 기존 플라즈마 Cash에서 지적되었던 문제점들을 개선하였다.

2.1.2 일반 상태 변화(General State Transition)

2.1.2.1 플라즈마 Leap

일반 상태에 관한 연산(General Computation) 기능을 비트코인의 Pay-to-Script-Hash (P2SH)와 유사한 사용 조건(Spending Condition)이라는 일종의 작은 프로그램으로 구현하였고, 상태(State) 모델을 사용하기 위해 스토리지 루트해시를 관리하는 기능을 가진 Non-fungible Storage Token(NST)를 제안했다[3]. NST를 통해 기존의 More Viable Plasma에서 사용되는 탈출 모델(Exit Model)[27]을 그대로 사용할 수 있게 됨으로써 데이터 가용성 문제를 해결하였다.

2.1.2.2 플라즈마 EVM

플라즈마 EVM은 플라즈마 체인에서 EVM을 실행할 수 있는 모델이다. 상태 변화(State Transition)를 검증하기 위한 방법으로 트루빗 스타일의 검증게임(Truebit Like Verification Game)을 사용하고, 컨트랙트 계정(Contract Account; CA)의 탈출에 관한 권한과 방법(Exit Authority)을 정의하기 위해 요청 컨트랙트(Requestable Contract)라는 개념을 제시[13]하였다. 또한 데이터 가용성 문제 해결을 위해 *Continuous Rebase*를 고안했다. 플라즈마EVM은 토파막 네트워크의 코어로 사용된다.

2.1.3 플라즈마 상태 검증(State Verification)

2.1.3.1 트루빗 프로토콜(TrueBit Protocol)

트루빗은 검증자의 딜레마(Verifier's Dilemma)를 해결하기 위해 등장했다. 검증자의 딜레마란 복잡한 연산 검증이 어려워 채굴이 될 가능성성이 낮아지는 것을 말한다. 트루빗은 검증 절차를 별도로 두는 방식을 통해 온체인(On-Chain)의 트랜잭션 수수료를 절약해 검증자의 딜레마를 해결[37]하며, 복잡한 연산을 체인 밖(Off-Chain)에서 수행하고 결과만을 메인 체인에 올린다. 이때 사용된 방식이 검증게임(Verification Game)이다. 토파막 네트워크는 상태검증 절차로 트루빗 스타일의 검증게임(Truebit-Like Verification Game)을 응용한다.

¹Fixed Denomination: 고정 액면가액. Plasma Cash에서는 각 코인이 마치 현금처럼 고정된 액면가로만 사용이 가능하다.

²일반적인 Merkle Tree처럼 노드를 단순히 해싱하는 것이 아니라, 여기에 더해 각 노드의 Balance의 합을 해싱하여 저장하는 자료구조이다.

2.2 퍼블릭 블록체인의 트랜잭션 수수료 모델

2.2.1 이더리움(Ethereum)

이더리움은 이더리움 자원을 부정하게 사용하는 것을 방지하기 위해 모든 연산에 수수료를 부과한다[41]. 즉, 이더리움 블록체인에서 이뤄지는 모든 연산은 “가스”라는 수단으로 시·공간적 비용을 지불한다[23]. 이더리움의 모든 트랜잭션은 가스 제한(Gas Limit)과 가스 가격(Gas Price) 필드를 가지고 있다. 여기서 가스 제한이란 해당 트랜잭션이 얼마나 많고 복잡한 연산을 수행하느냐에 대한 예산(Budget)의 성격을 가지고 있고, 가스 가격은 트랜잭션을 블록에 포함시키기 위해 마이너들에게 지불하는 입찰가로서의 성격을 가지고 있다.

만약 트랜잭션이 수행되는 과정에서 소모한 가스 사용량(Gas Used)이 설정된 가스 제한(Gas Limit)보다 높을 경우, 이 트랜잭션은 유효하지 않은(Invalid) 트랜잭션이 된다. 이에 대해 가스 가격(Gas Price)이 (블록에 포함되기를 대기하는 다른 트랜잭션에 비하여 상대적으로) 충분히 높지 않은 경우 마이너들은 해당 트랜잭션을 블록에 포함시키지 않을 가능성이 높아진다.

중요한 점은 트랜잭션을 생성하는 계정(Sender, Transactor)이 가스 사용량(Gas Used) * 가스 가격(Gas Price) 이상의 이더를 트랜잭션 생성간 매번 지불해야 하고, 이를 위해 일정량의 잔액수준을 상시 유지해야 한다는 것이다. 이더리움 블록체인은 이러한 형태의 수수료 모델을 통해 서비스 거부공격(DoS Attack)을 방지하고 “블록체인의 상태 변화에 대한 권리”라는 한정된 자원에 대한 분배문제를 해결했다. 하지만 사용자(Transactor)가 트랜잭션 실행에 대해 일일이 비용을 부담하는 방법으로 이더리움 블록체인을 통해 만들어지는 탈중앙화 어플리케이션(DApp)들의 사용성을 철저히 떨어뜨렸다. 이에 불편을 느낀 블록체인 개발자들은 수수료 없는 체인을 요구하게 되었고, 이런 맥락에서 스팀과 이오스가 등장했다.

2.2.2 스팀(Steem)

대부분 블록체인에서 DoS 공격을 방지하기 위해 트랜잭션마다 수수료를 지불하는 반면, 스팀(Steem)은 계정에 대역폭(Bandwidth)을 두어 무분별한 트랜잭션이 생성되는 것을 방지[36]한다.

스팀의 네트워크 혼잡도에 따라 유저 1인당 허용되는 트랜잭션 대역폭은 유동적으로 조절된다. 스팀 계정 1개당 스팀 파워(Steem Power)라는 별도의 토큰을 활용하여 스팀에 기여하는 가중치를 책정하고 스팀 파워 보유량에 따라 각 계정마다 할당받을 수 있는 대역폭이 달라지게 된다. 하지만, 이 또한 해당 유저가 토큰을 소유하고 있어야만 대역폭을 할당받을 수 있기 때문에, 대역폭을 위임하는 등의 다양한 기능을 지원하지 못한다는 점에서 한계가 있다.

2.2.3 이오스(EOS)

이오스의 경우, 블록체인의 엔드 유저는 블록체인 네트워크를 관리하는 21명의 BP로부터 Bandwidth, CPU, RAM이라는 3가지의 컴퓨팅 리소스를 할당받아 블록체인을 사용할 수 있다[5]. 이오스 블록체인에서 트랜잭션을 만들 때 필요한 리소스는 Bandwidth와 CPU로서, 이오스(EOS)의 보유량에 따라 트랜잭션을 보내는 계정에 의해 부담되고, 계정간 리소스 임대도 가능하다. RAM은 시장에서 거래가 가능한 자산이고 블록체인에 기록할 데이터를 저장하는 Storage로서 기능한다.

이오스 트랜잭션 수수료 모델의 장점은 수수료를 지불하는 유저의 입장에서 단순히 지불되어 없어지는 비용이 아니라, 각 자원들(Network, CPU, RAM)을 예치(Stake)한 만큼 일정량의 트랜잭션을 지속적으로 발생시킬 수 있다는 점이다. 때문에 이오스 어플리케이션은 이를 사용하는 유저들에게 블록체인 사용간 트랜잭션 수수료가 없는 것과 같은 사용자 경험을 만들 수 있다. 즉, EOS를 한 번 예치하기만 하면 자신에게 할당된 블록체인 리소스의 범위 내에서는 자유롭게 사용할 수 있도록 만들 수 있다.

다만, 이오스 블록체인에서 계정을 생성하기 위해서는 RAM이 필요하고 RAM은 시장에서 거래되는 자산이기 때문에 계정을 생성할 때 비용이 수반된다. 이는 시빌 공격(Sybil Attack)을 막기 위한 방법으로서는 훌륭할 수 있으나, 유저의 진입 장벽을 높이는 요소이고 RAM의 가격 변동성[38] 또한 블록체인의 원활한 사용을 방해 할 수 있다.

3장 시스템 구성

3장은 토파막 플라즈마에서 활용되는 플라즈마EVM[13], 트랜잭션 수수료 위임모델[25]등 다양한 기술의 핵심을 요약하고 있다. 플라즈마EVM은 기존의 플라즈마와 마찬가지로 플라즈마 체인(자식체인)의 상태를 주기적으로 이더리움 메인 체인에 요약 및 제출하고, 이 과정에서 플라즈마 체인 운영자가 올바르지 않은 방식으로 블록을 생성한 경우 검증게임을 통해 블록에 대한 올바름을 검증¹한다. 검증게임의 승패 여부에 따라 플라즈마 체인 오퍼레이터 혹은 챌린저(Challenger)에게 불이익을 주고, 이것으로 플라즈마 체인의 안전성을 높인다. 더해서 오퍼레이터가 고의적으로 블록의 내용을 숨기려고 (Withholding)할 경우, Continuous Rebase를 이용해 데이터 불가용 상황을 극복할 수 있다.

토파막에서 사용되는 플라즈마EVM은 이더리움 가상 머신(Ethereum Virtual Machine)을 사용하기 때문에 기존의 이더리움 네트워크에 만들어진 많은 탈중앙화 어플리케이션(DApp)들을 플라즈마 체인으로 옮겨 성능을 개선[35]시킬 수 있다. 더하여 토파막 플라즈마에서 트랜잭션을 생성하는 사용자는 제3의 계정에게 수수료를 위임[25]할 수 있다. 이것을 활용해, 초기 맵을 운영하는 DAO는 단일 혹은 다수의 수수료 위임계정을 만들어 서비스 이용자에게 “수수료가 없는 듯한 유저 경험(UX)”를 제공할 수 있다.

3.1 일반 상태 변화 강제(General State Transition Enforcement)

3.1.1 일반 상태의 개념과 연산 강제

상태(State)란 어떠한 시스템의 상태가 변환(Transition)되기 위해 대기하는 것을 말하며, 상태 변화(State Transition)이란 일련의 행위 집합(Set of Actions)이 특정한 조건을 만족하거나 어떠한 이벤트가 발생했을 때 해당 상태가 바뀌는 것을 뜻한다[20]. 블록체인은 상태 변화 시스템(State Transition System)[6]이고, 만약 이 시스템에서 변환(Transition)이 가능한 무한대의 행위 집합과 이로 인해 만들어지는 상태 결과를 정의할 수 있다면 이는 튜링 완전(Turing Complete)에 가까워 진다. 일반상태(General State)란 튜링 완전에 가까운 시스템의 (무한한) 상태 집합을 지칭한다.

튜링 완전한 시스템은 또 다른 튜링머신 혹은 상태 이전 시스템을 모델링할 수 있다. 예를 들면 시스템A에 시스템B 자체를 프로그래밍 해 두고 이 프로그램의 입력값과 산출값을 바탕으로 시스템B를 평가하면, 시스템B가 수행한 특정 상태 변화 함수(State Transition Function)가 올바른지 혹은 올바르지 않은지 여부를 판단할 수 있다².

¹구체적으로 트루비트의 검증게임(Truebit-Like Verification Game) 방식을 이용한다

²중요한 점은 모델링의 대상이 되는 시스템이 튜링 완전할 경우, B시스템의 평가가 이뤄지는 시점에 데이터 가용성 문제가 발생하면, 해당 시스템의 상태 변화를 검증하는 것이 불가능해진다. Johann Barbie의 Why Smart Contracts are NOT feasible on Plasma[4]는 이러한 상황을 잘 묘사하고 있다.

3.1.2 요청 가능 컨트랙트 : 진입과 탈출 규칙의 정의(Requestable : Defining Entering and Exiting Rules)

이더리움은 일반 상태(General State)를 다루기 위해 스마트 컨트랙트라는 객체를 만들어 사용한다. 예를 들어 이더리움 메인넷에서 가장 많은 토큰(ERC20) 컨트랙트는 i) 다수의 계정과 ii) 각 계정이 보유한 잔액 상태, iii) 잔액의 변경 규칙을 정의한 프로그램(객체)으로 볼 수 있다. 토큰(ERC20)의 전송 함수는 전송자의 잔액 상태가 줄어들고, 수신자의 잔액 상태가 늘어나는 로직으로 정의된다.

진입(Enter)이란 앞서 스마트 컨트랙트에 정의한 변수의 상태 변화 규칙에 모순되지 않는 방식으로 해당 변수를 루트체인에서 플라즈마 체인으로 옮기는 것을 뜻하고, 탈출(Exit)이란 그 반대의 상황을 뜻한다. 이더리움의 상태 변수(State Variable)와 그 변경 요청(Apply Request Function)을 플라즈마 체인에 반영(Reflect)하기 위해서는 목적하는 변수에 맞는 진입(Enter)과 탈출(Exit) 로직을 구현해야 한다. 예를 들어 이더리움 메인 체인에 있는 토큰의 잔액 상태를 플라즈마 체인으로 옮기기 위한 진입(Enter)에 관한 규칙은 다음과 같다.

1. 이더리움 메인 체인에서 플라즈마 체인으로 반영(Reflect) 시킬 대상 변수(잔액)가 기록된 위치와 진입시킬 토큰 수량을 정한다.
2. 이더리움 메인 체인의 잔액 중 옮길 대상이 되는 토큰을 소거(Burn)한다.
3. 플라즈마체인으로 옮길 대상이 되는 토큰을 발행(Mint)한다.

진입 규칙은 다양하게 정의할 수 있는데, 예를 들어 2번 규칙은 소거(Burn)가 아닌 동결(Freeze)로 구현될 수도 있다. 그리고 이러한 진입과 탈출 규칙의 공통 명세를 공유하는 컨트랙트 구현체를 요청가능한 컨트랙트(Requestable Contract)라 한다.

다음의 [숫자 카운터](#)와 [토큰](#)은 매우 단순한 형태의 스마트 컨트랙트로 진입과 탈출 규칙들에 이해의 폭을 넓혀줄 것이다[30][24].

3.1.2.1 숫자 카운터(Number Counter)

숫자 카운터는 숫자를 세는 프로그램으로, 기록된 수는 0부터 1씩 커진다. 최초의 숫자 카운터는 이더리움 메인 체인에서 동작하다가 특정한 시기를 기준으로 플라즈마 체인으로 옮겨서 사용될 수 있고 그 반대도 가능하다. 중요한 점은 이렇게 상태를 주고받는 과정에서 두 체인이 구조적으로 분리되어 있음에도 불구하고, 이 숫자 카운터가 관리하는 숫자 상태에 대한 전역 값은 논리적으로 문제가 없어야 한다는 것이다.

다음의 기본 카운터(BaseCounter) 컨트랙트 코드는 기초적인 카운터의 구현체로써, 이어지는 단순 카운터(Simple Counter)와 동결되는 카운터(Freezable Counter)는 아래의 컨트랙트를 상속하여 구현되어 다양한 방식의 요청 가능한(Requestable) 카운터로 변조될 수 있다.

```
contract BaseCounter {
    uint n;
    function count() external {
        n++;
    }
}
```

```

function getCount() external view returns (uint) {
    return n;
}
}

```

사용자는 count()함수를 호출해서 카운터의 값을 1씩 늘릴 수 있고, 특정한 시점에 카운터 값을 알고 싶다면 getCount()함수를 호출해 현재의 값을 확인할 수 있다.

3.1.2.1.1 단순 카운터(Simple Counter) 누구나 숫자를 증가시킬 수 있는 카운터 컨트랙트가 있을 때, 가장 직관적인 진입 규칙은 이더리움 메인 체인에서 숫자를 감소시키고 토파크 플라즈마 체인에서 그만큼 값을 증가시키는 것이다. 탈출 규칙은 반대로 토파크 플라즈마 체인에서 먼저 값을 감소시키고 이더리움 메인 체인에서 값을 증가시킨다. 이를 구현한 단순 카운터(Simple Counter)의 실행 흐름을 도식하면 다음과 같다.

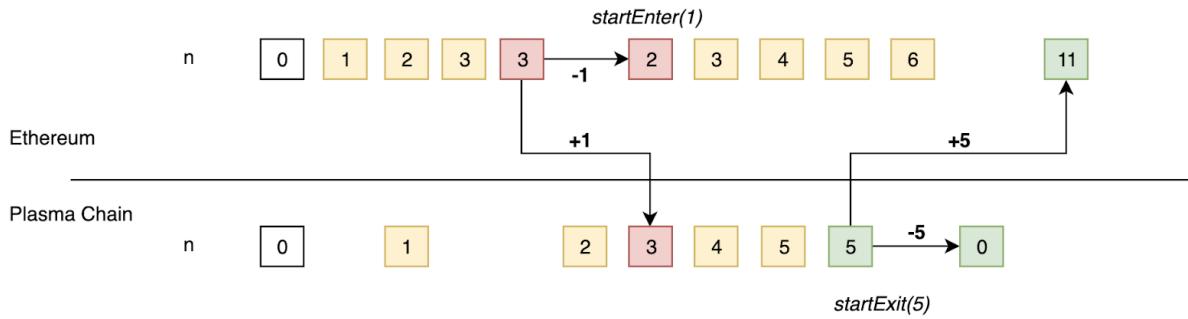


Figure 3.1: 단순 카운터(Simple Counter)

위 그림은 각 체인에서 컨트랙트 변수 `n`이 변경되는 모습을 묘사하고 있다. 노란색 네모는 count() 함수의 호출, 빨간색은 진입, 초록색은 탈출로 인해 변경된 것을 의미한다. 이더리움 메인 체인에서의 카운터 값의 상태 변화 규칙을 명세한 applyRequestInRootChain() 함수의 핵심 코드는 다음과 같다.

```

function applyRequestInRootChain (...) {
    ...
    if (isExit) {
        n = n.add(trieValue.toUInt());
    } else {
        n = n.sub(trieValue.toUInt());
    }
    ...
}

```

이더리움 메인 체인에서 플라즈마 체인으로 진입하는 경우에는 이더리움 메인 체인 카운터의 값이 줄고, 플라즈마 체인에서 이더리움 메인 체인으로 탈출하는 경우에는 값이 커지는 것을 볼 수 있다.

단, 단순 카운터(Simple Counter)는 변수 `n`이 진입과 탈출로 인해 감소할 수 있으므로, 사용자가 특정한 상태에 본인이 카운팅을 얼마나 했는지 파악하기 위해 이더리움 메인 체인과 토파크 플라즈마

체인 양쪽의 상태를 읽어와 그 값을 별도로 합하는 과정을 거쳐야 한다. 반면에 카운트 값의 진입, 탈출의 여부와 관계없이 양쪽 체인의 카운터를 동시에 사용할 수 있는 장점이 있다.

3.1.2.1.2 동결 되는 카운터(FreezableCounter) 다른 방식은 진입과 탈출이 일어나기 직전 각 체인에 있는 카운터 컨트랙트의 수를 동결 시키는 것이다. 동결되는 카운터(FreezableCounter)는 동결 후 요청 방식을 통해 숫자가 감소하는 경우는 피할 수 있다.

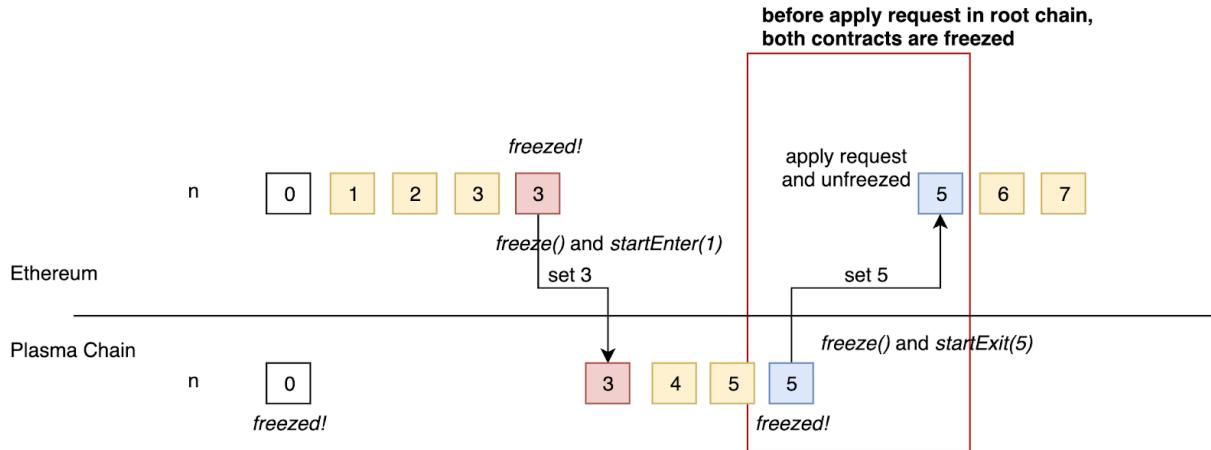


Figure 3.2: 동결되는 카운터(Freezable Counter)

이더리움 메인 체인의 상태 변화 규칙은 다음과 같다.(`applyRequestInRootChain` 함수의 핵심 구현 코드는 다음과 같다.)

```
function applyRequestInRootChain ( . . . ) {  
    . . .  
    require(frozen)  
    if (isExit) {  
        frozen = false;  
        n = trieValue.toInt();  
    } else {  
        require(n == trieValue.toInt());  
    }  
    . . .  
}
```

이더리움 메인 체인에서 토파막 플라즈마 체인으로 진입이 이뤄지면 이더리움 메인 체인의 상태는 동결되고, 토파막 플라즈마 체인에서 이더리움 메인 체인으로 탈출이 이뤄지면 동결된 상태값이 풀린다. 이렇게 동결되고 풀리는 과정에서 양 체인은 카운터 상태값을 주고받는다. 이 구현의 장점은 이더리움 메인 체인과 토파막 플라즈마 체인 중 동결되지 않은 체인이 카운터 값에 대한 전역상태를 상시 관리하고 있다는 점이다. 따라서 사용자는 현재의 전역 카운터 값을 알기 위해 동결되지 않은 체인의 카운터값만을 읽으면 된다.

하지만 탈출이 이뤄진 후 양쪽 체인에 상태값이 반영되기 위해서는 챌린지 과정을 거쳐야 하는데, 이 기간동안 카운터 컨트랙트를 사용할 수 없는 단점이 있다. 이러한 상황을 방지하기 위해서는 진입에 사용되는 상태 변수와 탈출에 사용되는 상태 변수를 다르게 두어야 한다.

3.1.2.1.3 추적 가능한 카운터(Trackable Counter) 동결되는 카운터(Freezable Counter)는 챌린지가 끝날때까지 카운터를 사용할 수 없는 단점이 있다. 이러한 문제를 해결하기 위한 대안적인 구현으로 추적 가능한 카운터(Trackable Counter)를 생각 해 볼 수 있다. 추적 가능한 카운터는 진입과 탈출 과정에 별도의 상태 변수 requestableN을 통해 진입 혹은 탈출이 가능한 상태인지 확인한 이후의 양 체인의 카운터 값을 증감시킨다.

루트체인에서 실행되는 applyRequestInRootChain() 함수의 핵심 로직은 다음과 같다.

```
function applyRequestInRootChain (...) {
    ...
    if (isExit) {
        n = n.add(_n);
    } else {
        requestableN = requestableN.sub(_n);
    }
    ...
}
```

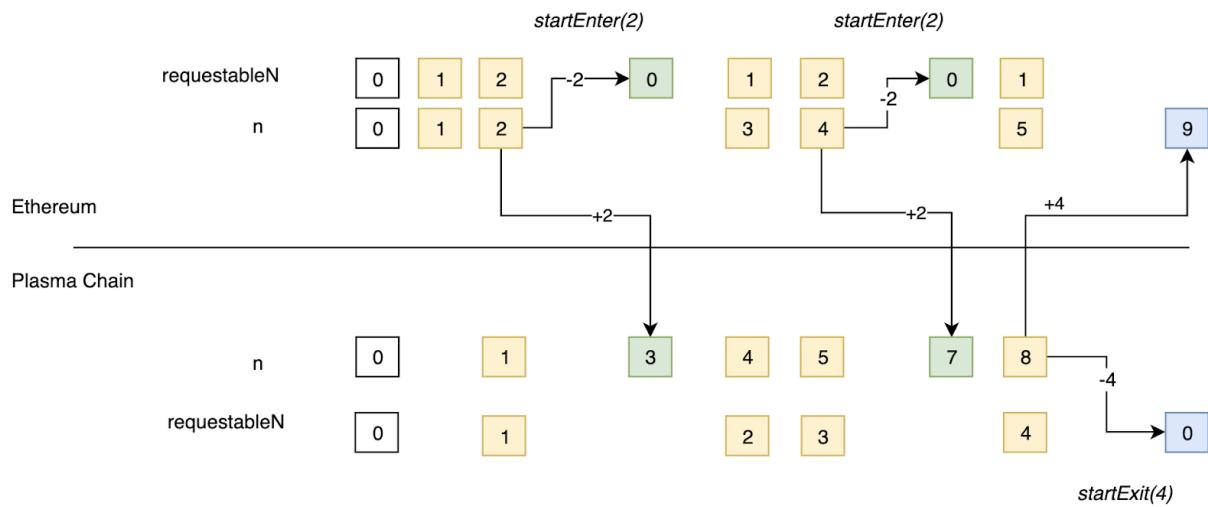


Figure 3.3: 추적 가능한 카운터(Trackable Counter)

3.1.2.2 토큰(Token)

토큰 컨트랙트의 경우 가장 단순한 형태의 진입(Enter) 규칙은 이더리움 메인 체인에서 토큰을 소거하고(Burn) 플라즈마 체인에서 토큰을 발행하는(Mint) 것이다. 반대로 탈출(Exit)은 플라즈마 체인에서 토큰을 소각하고 이더리움 메인 체인에서 토큰을 발행한다.

3.1.2.2.1 요청 가능한 단순 토큰(RequestableSimpleToken) 요청 가능한 단순 토큰(RequestableSimpleToken)은 OpenZeppelin의 상시 발행 가능한 ERC20 토큰(ERC20Mintable)을 요청 가능한(Requestable) 형태로 변형한 것이다. 해당 컨트랙트는 두 상태변수 owner, balances에 대한 진입과 탈출을 허용한다. 즉, 요청가능한 단순 토큰은 이더리움 메인체인과 플라즈마 체인 양쪽을 넘나들면서 사용될 수 있다.

루트체인에서 실행되는 applyRequestInRootChain() 함수의 핵심 로직은 다음과 같다.

```
// apply exit
if (isExit) {
    if (bytes32(0) == trieKey) {
        owner = requestor;
    } else if (keccak256(bytes32(requestor), bytes32(2)) == trieKey) {
        balances[requestor] += trieValue.toInt();
    }
} else {
    // apply enter
    if (bytes32(0) == trieKey) {
        // just check permission.
        require(owner == requestor);
    } else if (keccak256(bytes32(requestor), bytes32(2)) == trieKey) {
        require(balances[requestor] >= trieValue.toInt());
        balances[requestor] -= trieValue.toInt();
    }
}
```

trieKey에 따라 어느 변수에 대한 요청인지 식별할 수 있다. 이 때 trieKey가 0x00일 때에는 owner 변수에 대한 요청이라고 정의한다. balances변수는 address를 키로 가지는 mapping(hash table)이기 때문에, trieKey를 requestor에 대한 식별자로 사용하기 위해 sha3(requestor, 0x02)를 사용한다.

3.1.2.2 요청가능한 ERC20 래퍼(RequestableERC20Wrapper) 기존의 ERC20 토큰 인터페이스 자체는 요청가능(Requestable)하지 않기 때문에 이를 토카막 플라즈마에서 사용하기 위해서 새로운 토큰으로 감싸는(Wrapping) 작업이 필요하다. 이렇게 감싸진 토큰은 원본 토큰에 대한 회수권(Redemption)이 된다. 토카막 플라즈마 체인에서는 이 회수권을 사용할 수 있고, 메인체인에서 이를 이용해 언제든지 이미 만들어진 토큰으로 교환할 수 있다.

다음의 RequestableERC20Wrapper 컨트랙트는 해당 기능을 구현하고 있다.

```
contract RequestableERC20Wrapper is StandardToken, RequestableI {

    function deposit(uint _amount) external isInitialized returns (bool) {
        mint(msg.sender, _amount);
        require(token.transferFrom(msg.sender, this, _amount));
        return true;
    }
```

```

    }

    function withdraw(uint _amount) external isInitialized returns (bool) {
        burn(msg.sender, _amount);
        require(token.transfer(msg.sender, _amount));
        return true;
    }

}

```

다음은 기준이 되는 ERC20 토큰 컨트랙트를 입금, 출금하는 기능을 구현한다.

```

function applyRequestInRootChain(
    bool isExit,
    uint256 requestId,
    address requestor,
    bytes32 trieKey,
    bytes trieValue
) external isInitialized returns (bool success) {
    ...
    uint v = decodeTrieValue(trieValue);

    if (isExit) {
        mint(requestor, v);
    } else {
        burn(requestor, v);
    }
    return true;
}

function applyRequestInChildChain(
    bool isExit,
    uint256 requestId,
    address requestor,
    bytes32 trieKey,
    bytes trieValue
) external returns (bool success) {
    ...
    uint v = decodeTrieValue(trieValue);
    if (isExit) {
        burn(requestor, v);
    }
}

```

```

    } else {
        mint( requestor , v );
    }
    return true;
}

```

진입 요청의 경우, 이더리움 메인 체인에서 토큰을 소각하고 토파크 플라즈마 체인에서 토큰을 발행한다. 반대로 탈출 요청은, 토파크 플라즈마 체인에서 토큰을 소각하고 이더리움 메인 체인에서는 토큰을 발행한다. 이 때 기존 ERC20 토큰은 이더리움 메인 체인의 요청 가능한 ERC20 래퍼 컨트랙트(RequestableERC20Wrapper)로 감싸두었기 때문에 사용자는 항상 이더리움 메인 체인에서 감싼 토큰으로 원본 토큰을 회수 할 수 있다.

3.1.3 검증게임 : 챌린지(Verification Game : Challenge)

튜링 완전한 시스템은 다른 튜링 완전한 시스템을 모델링 할 수 있다. 검증자는 이더리움 메인체인에 만들어지고, 플라즈마 블록체인의 상태 변화 규칙을 모델링한다. 루트체인 컨트랙트는 검증자 컨트랙트를 이용해 플라즈마 블록체인의 상태 변화가 올바르게 일어났는지 판단할 수 있다. 유저들은 플라즈마 블록의 내용을 지켜보면서 올바르지 못한 블록이 생성되는걸 감시하고, 문제를 식별할 경우 챌린지(Challenge) 통해 이를 정정한다. 챌린지의 대상은 이중지불연산(double spending computation)뿐만 아니라, 프로토콜에서 정의한 규칙 외의 행위가 만들어낸 결과를 대상으로도 이뤄질 수 있다. 널-계정 챌린지, 탈출 챌린지와 [최소가스가격](#) 챌린지는 그러한 상황을 예시하고 있다.

3.1.3.1 검증자와 검증(Verifier and Verify)

이더리움의 상태 변화는 이더리움 가상 머신(Ethereum Virtual Machine; 이하 EVM)에 의해서 일어나기 때문에, 이더리움 가상머신을 사용하는 플라즈마 체인의 상태검증을 위해서는 EVM 자체를 루트체인에 모델링해야 한다. solevm³은 이더리움의 스마트 컨트랙트 언어인 솔리디티(Solidity)로 구현된 EVM이며, solEVM-enforcer⁴등을 통해서 스마트 컨트랙트의 기능적 완성도를 높이는 과정에 있다. 향후 플라즈마가 [현재의 EVM보다 더 다양한 기능을 필요로 한다면](#), 그에 맞춰 검증자 컨트랙트는 다양해질 수 있다. 대표적인 예로 이더리움 2.0이 도입되는 과정에서 eWASM을 통해 opcode(이더리움 기본 연산 단위) 구성과 실행모델은 변경될 수 있고, eWASM 실행 모델에 맞는 검증자 구현으로 새로운 상태 변화 규칙이 적용된 플라즈마 체인을 검증할 수 있다.

토파크 플라즈마의 블록들은 항상 올바르게 체인의 상태를 변환(Transition) 시켜야한다. 예를 들어 사용자 A가 사용자 B에게 토큰을 10단위 전송했을 때 사용자 A의 잔액이 10단위 감소하고 사용자 B의 잔액이 10단위 증가하는 것이 올바른 연산이다. 만약 두 사용자의 잔액증감이 이와 다르다면 올바른 연산이 진행되지 않은 것이다.

³<https://github.com/Ohalo-Ltd/solevm>

⁴<https://github.com/leapdao/solEVM-enforcer>

3.1.3.2 탈출 챌린지(Exit Challenge)

탈출 챌린지는 플라즈마 체인에서 탈출 요청이 실패하였을 경우 이를 이더리움 메인 체인 컨트랙트에 제보함으로써 올바르지 않은 탈출 요청이 이더리움 메인 체인에도 반영되지 않도록 방지한다[13]. 예를 들어 특정한 계정에 토큰 잔액이 100단위 임에도 불구하고 1,000단위 토큰을 탈출시키는 요청이 만들어 질 수 있다. 이 경우 해당 탈출 요청 트랜잭션의 실행은 취소(Revert)된다. 따라서 이를 바탕으로 이더리움에서 해당 요청이 정당한 요청인지 판단할 수 있다.

단, 탈출 챌린지가 성공하더라도 체인 예치금은 삭감되지 않는다. 대신 유저는 탈출 요청을 하는 과정에서 소량의 탈출 보증금(Exit Deposit)을 예치하고, 탈출 챌린지가 성공할 경우 해당 보증금이 챌린저에게 부여되어 탈출 챌린지에 대한 경제적 유인[22]으로 쓰이게 된다.

3.1.3.3 프로토콜 챌린지 : 널-계정, 탈출, 최소가스가격(Protocol Challenge : Null-Address, Exit and MGP)

토카막 플라즈마(플라즈마EVM)의 널-계정은 플라즈마 체인 외부에서 발생한 상태 변환 요청을 반영하는 역할을 수행한다. 만약 오퍼레이터가 요청 트랜잭션을 요청 블록이 아닌 다른 블록(비요청블록)에 포함할 경우 널-계정 챌린지의 대상이 된다[13]. 이를 챌린저가 확인하게 되면 해당 블록에 포함된 트랜잭션 데이터를 이용하여 챌린지를 하게 된다.

Minimum Gas Price은 토카막 플라즈마에서 최소한으로 지출되어야 하는 가스가격의 한도를 의미한다. 만약 플라즈마 오퍼레이터가 최소가스가격 이하를 지불하는 트랜잭션을 플라즈마 블록에 포함시킬 경우 해당 블록은 챌린지의 대상이 된다.

토카막 플라즈마는 이더리움2.0의 로드맵이 진행되는 과정에서 더욱 개선될 가능성이 있으며, 프로토콜 챌린지는 향후 만들어질 다양한 추가 규칙과 이에 대한 위반행위를 제지할 수 있는 유용한 수단으로 활용 가능하다.

3.2 데이터 가용성 문제(Data Availability Problem)

플라즈마에서 데이터 가용성 문제란 플라즈마 체인의 사용자가 블록 데이터에 접근하지 못하는 문제를 의미한다. 이는 오퍼레이터가 악의적으로 블록을 전파하지 않는 블록 인질 공격(Block Withholding Attack)이나 네트워크 장애로 인해 발생할 수 있다.

데이터 가용성 문제가 해결되지 않는다면 플라즈마 데이터는 안전(Security)을 보장받을 수 없다. 대부분의 플라즈마는 오퍼레이터 혹은 사용자가 이중지불을 시도할 때, 이에 대한 증거를 루트체인에 제출하여 이중지불을 막는 챌린지 시스템을 두고 있다. 하지만 챌린지의 대상이 되는 데이터에 누구도 접근할 수 없는 경우, 데이터의 유효성(Validity)을 판단하는 것이 불가능해진다. 즉, 데이터 가용성에 문제가 발생하면 챌린지 시스템도 정상적으로 동작하기 어려워지고 결과적으로 플라즈마 체인의 안전은 보장받을 수 없다.

모든 플라즈마는 데이터 가용성 문제의 진위여부를 명확하게 판별하는 대신, 사용자가 언제든지 안전하게 해당 체인에서 탈출할 수 있도록 보장하고자 한다. 중요한 점은 튜링 불완전한 플라즈마는 비교적 손쉽게 사용자들에게 항상 안전한 탈출을 보장해주는 것이 가능한 반면, 토카막 플라즈마와 같은 튜링 완전한 플라즈마는 안전한 탈출을 보장하는 것이 쉽지 않다는 점이다.

이유는 블록 인질 공격(Block Withholding Attack)이 시작되면 상태 변화를 검증하는 것이 불가능하기 때문에 모든 상태가 유효하지 않은 것으로 간주되어야 하기 때문이다.^[4] 즉, 사용자는 데이터 가용성 문제가 발생한 이후에 탈출하더라도 이미 모든 상태는 유효하지 않기 때문에 안전을 보장받지 못하게 되는 것이다. 때문에 대부분의 플라즈마는 튜링 완전성을 제한하여 이러한 문제를 피해간다.

토카막 플라즈마는 플라즈마EVM의 Continuous Rebase[13]를 통해 플라즈마 체인의 기능을 제한하지 않고 데이터 가용성 문제를 해결한다. 토카막 플라즈마에서 데이터 가용성 문제가 생길 경우 사용자는 비상탈출 요청(Escape Request)을 제출하여 해당 체인에서 탈출할 수 있다. 비상탈출 요청은 항상 최종적으로 확정된 상태(Last Finalized State)를 기준으로 우선 처리되기 때문에 가용성 문제가 발생하더라도 사용자의 데이터는 항상 안전을 보장받을 수 있다.

3.3 무합의성(No Consensus)

합의가 없는 것은 그 어떤 합의 알고리즘보다 빠르다(No consensus is faster than no consensus). 가장 단적인 예로 권위 증명(Proof of Authority; PoA)을 사용하는 노드로 구성한 블록체인의 성능은 노드가 자리잡은 물리적인 하드웨어와 네트워크 스펙에 의해 결정된다. 이는 일반적인 퍼블릭 블록체인에서 수백, 수천의 불특정 다수 노드가 참여하는 작업증명(Proof of Work)과 지분 증명(Proof of Stake)이 목표로 하는 성능과는 비교할 수 없을 정도로 빠르다.

플라즈마 체인의 안전성(Security)은 플라즈마 프로토콜에 의해 루트체인(이더리움 메인체인)이 보장한다. 따라서 안정성 확보를 목적으로 별도의 합의 알고리즘을 플라즈마에 도입하는 것은 불필요하며, 플라즈마 체인에서 예치금을 확보한 노드만이 블록을 생성하는 PoA(Proof of Authority)를 사용하는 것이 권장된다. 하지만 토카막 플라즈마가 PoS(Proof of Stake), BFT(Byzantine Fault Tolerance) 류의 합의 알고리즘을 택한다 해도 플라즈마 네트워크 구성에는 제한이 없다⁵. 단, 플라즈마 체인이 별도의 합의 규칙에 의해 블록을 생성한다 하더라도, 합의를 통해 만들어내는 상태 변화 결과 자체는 부모체인에서 검증자가 규정한 연산 규칙 범위 내에서 이루어져야 한다.

정리하면, 토카막 프로토콜 자체는 플라즈마 체인에 특정한 합의를 요구하지 않는다. 이러한 무합의성은 각각의 플라즈마 체인의 존재 목적과 상황에 따라 다양한 합의 알고리즘을 도입할 수 있는 선택지를 주어 유연한 플라즈마 체인 구성을 가능하게 한다.

⁵이러한 형태의 별도의 합의 알고리즘 도입은 POA 알고리즘을 사용하는 오퍼레이터의 막강한 검열 권한을 약화시켜, 플라즈마 체인이 검열 저항성을 갖도록 할 수 있다.

4장 경제 모델(Economic Model)

4.1 토큰과 주조차익(Token and Seigniorage)

4.1.1 토큰의 사용과 단위

토카막 네트워크는 톤(TON)이라는 ERC20¹ 토큰을 가지고 있으며, 톤(TON)은 i) 플라즈마 체인의 올바른 운영 및 ii) 플라즈마 체인에 대한 시빌 공격(Sybil Attack) 방지, 두 가지 목적으로 각각 a) 체인 예치금(Chain Staking) 및 b) [플라즈마 트랜잭션 수수료](#)로써 활용된다. 토카막 네트워크 내에서 플라즈마 체인을 열기 위해서 오퍼레이터는 이더리움 메인 체인에 최소한의 톤(TON)을 예치해야 한다. 더하여 만약 플라즈마 체인이 챌린지(Challenge) 된다면 이 예치금은 챌린저의 상금이 된다. 또한 톤(TON)은 토카막 프로토콜에 의해서 만들어진 플라즈마체인에서 트랜잭션을 발생시키는데 지불되는 수수료로 활용되는데, 이 과정에서 [스테미나\(Stamina\)](#)라는 독특한 형태로 변환되게 된다.

일반적으로 ERC20 토큰은 10^{18} 소수점(Decimal)을 사용하는데, 낮은 단위로 토큰을 이용할 때 발생할 수 있는 혼란을 줄이기 위해서 각 단위에 대한 명칭은 다음으로 정한다.

- 1 = 푼(pooh)
- 10^{18} = 톤(ton)

이러한 명칭은 이더리움에서 “이더(Ether)”와 “웨이(Wei)”의 관계와 유사하며, 작은 단위인 푼(Poon)은 소액 결제나 기술적인 논의를 위해 활용되고 큰 단위인 톤(TON)은 일반적인 트랜잭션 및 큰 단위 거래에 사용될 것이다.

4.1.2 토큰 주조차익(Token Seigniorage)

토큰의 주조차익은 해당 플랫폼에서 가장 큰 부가가치를 발생시키는 행위에 대한 인센티브의 목적으로 활용되는 것이 바람직하다. 토카막 플라즈마에서 가장 중요한 행위는 플라즈마 블록을 만들어내서 트랜잭션 처리 부담을 줄이는 것이다. 따라서 새롭게 발행되는 톤(TON)은 플라즈마 체인을 운영하는 오퍼레이터(마이너)들이 플라즈마 블록을 이더리움 메인체인에 커밋할 때 주어진다.

이렇게 새롭게 발행되는 톤(TON)은 초기 발행량(Initial Supply)의 0.25 배수가 매년 영구히 신규 발행된다(고정 %가 아닌, 고정 수량). 고정 수량을 무한히 발행하는 방식은 이미 이더리움에서 사용하는 토큰 발행 모델이며[6], 이를 통해 톤(TON)을 시장이 아닌 플라즈마 체인 운영을 통해서 얻을 기회를 장기간 열어줄 수 있다.

¹<https://en.wikipedia.org/wiki/ERC-20>

이 경우 토큰의 공급 성장률은 첫해 25%에서 장기적으로 0%에 가깝게 수렴하게 되는데, 과정에서 기여도 높은 플라즈마 체인의 오퍼레이터들과 커뮤니티의 영향력은 점차 확장될 수 있다.

4.1.3 주조차익 분배(Seigniorage Distribution)

오퍼레이터에게 주어지는 토큰 주조차익의 크기는 체인 예치금의 크기와 진입 예치금에 의해서 결정[22]된다. 체인 예치금이란 토카막 플라즈마 체인의 제네시스 블록이 생길 때 이더리움 메인체인에 예치하는 톤(TON) 토큰이며, 진입 예치금은 플라즈마 체인이 운영되는 과정에서 사용자가 플라즈마 체인으로 진입시킨 톤(TON) 토큰의 양이다.

만약 오퍼레이터가 높은 신뢰도를 바탕으로 플라즈마 체인을 운영하고자 한다면, 많은 양의 체인 예치금이 있을 것이고, 이를 통해 발생하는 톤(TON) 주조차익의 크기도 커질 것이다. 혹은 체인 예치금 자체는 높지 않더라도, 많은 진입 예치금이 있을 경우 오퍼레이터의 톤(TON) 주조차익은 커질 수 있다[22]. 다만 체인 예치금이 클수록 진입 예치금의 크기도 클 가능성이 높으며, 따라서 주조차익은 큰 자본을 보유한 오퍼레이터들에게 주로 주어질 가능성이 크다. 하지만 이 오퍼레이터들이 이중지불 등의 잘못된 블록(invalid block)을 만들어내면 예치금을 포함한 주조차익은 문제제기를 한 챌린저에게 분배된다[22].

주조차익의 핵심 원칙은 올바른 행위를 하는 주체에게 경제적 보상을 나누는 것이다. 토카막 네트워크는 올바른 플라즈마 블록을 만들어내는 오퍼레이터 및 올바르지 못한 오퍼레이터와 맞서 싸워 이긴 챌린저에게 보상을 나눔으로써 플라즈마 블록체인의 안전성을 보장하고자 한다.

4.1.4 이더(Ether)와 토카막 네트워크 토큰

토카막의 플라즈마는 블록의 요약본을 이더리움 메인체인에 기록하는 과정에서 토큰 주조차익이 생겨난다. 동시에 이더(Ether)는 플라즈마 블록이 메인체인에 커밋되는 과정에서 트랜잭션 수수료로 사용된다. 이때 이더(Ether)로 지불되는 커밋 트랜잭션 비용은 톤(TON)의 주조차익만을 목적으로 무분별하게 발생하는 트랜잭션을 억제하는 수단이 된다.

4.2 플라즈마 체인 예치금(Chain Staking)

토카막 플라즈마를 열기 위해서 오퍼레이터는 최소한의 톤(TON)을 이더리움 메인체인에 예치해야 한다. 예치된 톤(TON)은 오퍼레이터가 플라즈마 블록을 커밋하는 과정에서 받게 되는 토큰 주조차익의 근거로 쓰일 뿐만 아니라, 잘못된 블록을 커밋했을 때 발생할 수 있는 검증게임의 상금으로 쓰인다[22]. 즉, 오퍼레이터가 고의적으로 이중지불(Double Spending) 트랜잭션을 만들어 플라즈마 블록에 포함시킬 경우 챌린저가 일으킨 검증게임을 통해서 예치된 톤(TON)의 소유권이 이의제기를 한 챌린저에게 옮겨지게 된다. 예치된 톤(TON)은 마치 오퍼레이터가 향후에 발생할 분쟁상황(검증게임)을 대비해 의무적으로 마련해둔 소송비용 혹은 소송상금으로 볼 수 있고, 이 정책은 플라즈마 체인이 올바르게 유지되도록 유도하는 역할을 수행한다.

이렇게 예치된 플라즈마 체인 예치금은 사용자들이 플라즈마 체인을 선택하기 전에 체인의 안정성과 신뢰성을 판단할 수 있는 주요한 기준으로 활용될 수 있다. 많은 금액의 체인 예치금을 둔 플라즈마와 적은 금액의 톤(TON)을 예치금으로 쓰는 두 플라즈마를 비교한다면 더 많은 비용을 예비하는 쪽의

신뢰도가 높을 것이라고 직관적으로 판단할 수 있다. 다만 체인의 예치금과 오퍼레이터 자체의 신뢰도는 다를 수 있다. 만약 오퍼레이터가 블록체인 외부적으로 높은 신뢰도를 가지고 있다면 - 예를 들어 자본금이 매우 큰 기업이 직접 플라즈마 체인을 공개적으로 운영한다면 - 체인 예치금 수준은 낮아도 플라즈마 체인의 신뢰도는 높아질 수 있다.

그럼에도 불구하고 블록체인 상에서 신뢰를 높이기 위해서 오퍼레이터와 사용자는 더 큰 금액의 체인 예치금을 필요로하게 된다. 중요한 점은 체인 예치금으로 인해서 발생하는 편익은 오퍼레이터뿐만 아니라 사용자들까지 누리게 된다는 것이다. 하지만 체인 예치금을 마련하는쪽은 (일반적으로)오퍼레이터이고, 예치된 톤(TON)은 다른곳에 활용할 수 없으므로 기회비용이 발생한다. 체인 예치금을 단순히 겸중게임의 상금으로 활용하는 구조에서는 비용의 부담과 편익의 수혜 대상이 달라 공정하지 못한 자원분배가 이뤄지게 된다. 그렇기 때문에 체인 예치금의 기회비용을 보전하는 [토론의 주조차익](#)이 필요하다.

4.3 플라즈마 트랜잭션 수수료

4.3.1 시빌 공격과 트랜잭션 수수료(Sybil Attack and Transaction Fee)

사용자는 블록체인의 연산능력이라는 한정된 자원을 차지하기 위해 비용을 지불해야 하는데, 이더리움의 가스와 가스 가격은 그 자체로 “이더리움의 연산능력”에 대한 사용자의 지불 비용으로 볼 수 있다. 이러한 개념을 경제학에서는 경합성(Rivalry)과 배제성(Excludability)이라 부르며, “이더리움 블록체인의 연산능력”이라는 자원의 특징을 이 기준으로 정리하면 다음과 같다.

- 경합성(Rivalry) : 만약 한 계정이 다수의 트랜잭션을 만들어낸다면, 다른 계정은 트랜잭션에 포함되기 어렵다. 왜냐하면 블록에 담길 수 있는 트랜잭션 및 연산의 양이 정해져있기 때문이다 (이더리움의 경우 블록 가스 리밋).
- 배제성(Excludability) : 단위 연산을 수행하기 위해서 연산 단위 가스당 지출 하고자 하는 비용 (가스 가격)을 트랜잭션 생성 과정에서 결정한다.

배제성이 약한 자원은 공유자원의 비극(Tragedy of common resource)을 겪게 된다. 다르게 말하면 트랜잭션 수수료 자체가 없거나, 트랜잭션당 부과되는 수수료가 충분치 않은 경우 블록체인의 연산 자원이 과도하게 소비되는 문제가 발생한다. 이러한 의미에서 이더리움의 트랜잭션 수수료인 가스와 가스 가격은 사용자에게는 불편하지만 전체 네트워크의 안정성을 위해서 필수적인 요소임에 분명하다. 다만 이러한 수수료 부과방식은 다양할 수 있다. 이오스와 스팀등에서는 각 계정이 보유한 리소스에 따라서 일정한 기간동안 발생시킬 수 있는 권리를 부여하는데, 이러한 방식의 수수료 정책은 예측가능성 측면에서 유저 경험(UX)을 개선시킬 수 있다.

토카막 플라즈마는 수수료 위임 모델[25]을 바탕으로 유연한 트랜잭션 수수료 정책을 만들어 낸다. 앞으로 논의되는 [스테미나\(Stamina\)](#)와 [최소가스가격\(Minimum Gas Price; MGP\)](#)은 개선된 이더리움의 수수료 정책 개념을 구체적으로 기술하고 있다.

4.3.2 스태미나 : 수수료 위임과 재생

토카막 플라즈마 사용자는 톤(TON)을 거치해 플라즈마 체인에서 일정한 기간동안, 일정한 양의 트랜잭션을 일으킬 수 있는 권리인 “스태미나”를 만들 수 있다. 사용자가 만들어낸 트랜잭션은 -기존의 이더리움 메인체인이 계정의 이더리움 잔액을 차감하는 것과는 달리 - 스태미나를 소모하며, 소모된 스태미나는 일정한 기간이 지나면 재생된다[22]. 이러한 방식의 수수료 차감 정책은 마치 사용자가 일정한 기간 동안 플라즈마 블록체인을 사용할 수 있는 “대역폭(Bandwidth)”을 임차하는 것과 같으며, 임차비용은 해당 기간동안 스태미나 사용을 위해 거치해둔 톤(TON)의 시간 기회비용으로 볼 수 있다. 톤(TON)을 통해 만들어진 스태미나는 이를 거치한 사용자 본인의 계정 뿐만아니라, 다른 계정으로 빌려주는 것 또한 가능하다. 이렇게 만들어진 위임자-수임자 관계의 계정을 스태미나 쌍(Stamina Pair)이라 칭한다[26].

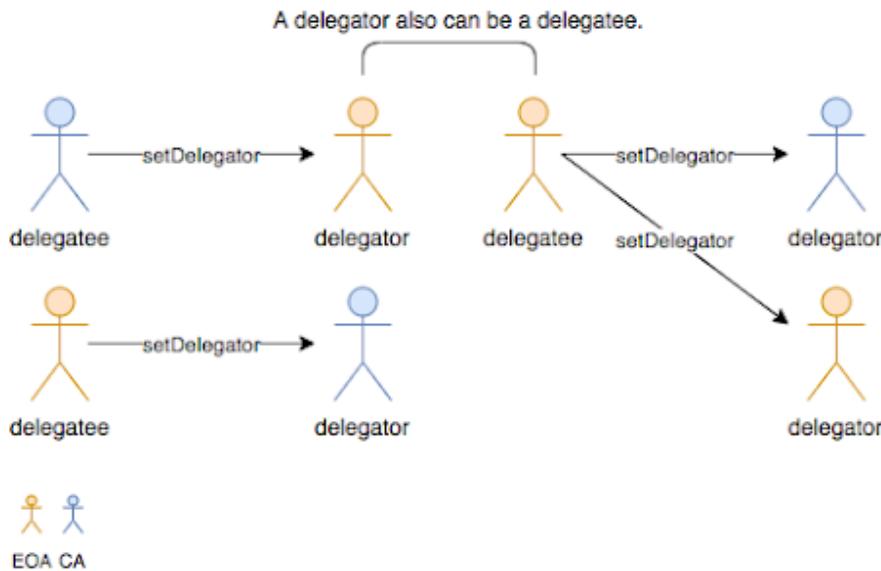


Figure 4.1: 스태미나 쌍(Stamina Pair)

스태미나 쌍은 다양하게 응용될 수 있다. 예를 들어 만약 서비스 제공자(DApp 개발팀)가 유저들에게 수수료의 존재를 인지하지 못하도록 만들고 싶을 경우, 서비스 제공자의 관리자 계정과 유저들의 계정을 스태미나 쌍으로 만들어 수수료를 대납 하는 것이 가능하다. 이는 마치 통신사가 선불폰을 구입한 고객들에게 일정한 기간동안 사용할 수 있는 무료 통화 시간을 나눠주는 것과 같으며, 이러한 방식으로 초기 고객 유치와 서비스 활성화를 유도할 수 있다. 마찬가지로 초기 사용자를 유치하고 활성화하기를 원하는 탈중앙화 어플리케이션 서비스 제공자들은, 스태미나 쌍을 적극적으로 활용할 가능성이 높다.

4.3.3 최소 가스 가격(Minimum Gas Price; MGP)

최소가스가격(Minimum Gas Price)이란 플라즈마 체인에서 트랜잭션 전송시 유저가 지불해야하는 최소한의 가스 가격을 의미한다. 토카막 플라즈마는 이더리움 메인체인과는 달리 높은 가스 가격(Gas

Price)을 지불하는 트랜잭션을 먼저 블록에 포함하는 정책을 택하지 않는다. 토카막 플라즈마의 오퍼레이터는 높은 가스가격을 지불하는 트랜잭션을 포함할 어떠한 유인도 존재하지 않기 때문이다(대신 오퍼레이터는 체인 예치금과 진입 예치금에 비례한 토큰 주조차익을 얻는다). 이러한 고정 수수료 정책이 가능한 이유는 대부분의 경우 블록이 가득차기 전까지 트랜잭션 수수료의 변동폭이 크지 않고 안정적으로 유지되기 때문이다[7].

최소가스가격은 플라즈마 블록체인이 만들어질 때 정해져야 하지만, 플라즈마 체인의 블록이 진행되는 도중에도 언제든 변경 가능하다. 이러한 변경 정책과 **스태미나 쌍**을 조합하면 매우 유연한 수수료 모델을 만들어 낼 수 있다. 예를 들어 최소가스가격을 매우 낮은 수준으로 설정하고, 서비스 제공자는 최소한의 톤(TON)을 확보해 유저들과 스태미나 쌍을 만든다면, 실질적으로 유저들과 서비스 제공자 모두 아무런 비용을 내지 않고 플라즈마 블록체인을 사용하게 된다. 하지만 이렇게 낮아진 비용은 **블록체인을 공유자원의 비극**에 빠뜨릴 수 있기 때문에 장기적으로 유지될 수 있는 것은 아니며, 시간이 지날수록 늘어나는 무임 승차자(Free Rider)를 막기 위해서 일정한 수준 이상으로 최소가스가격을 높이게 될 것이다.

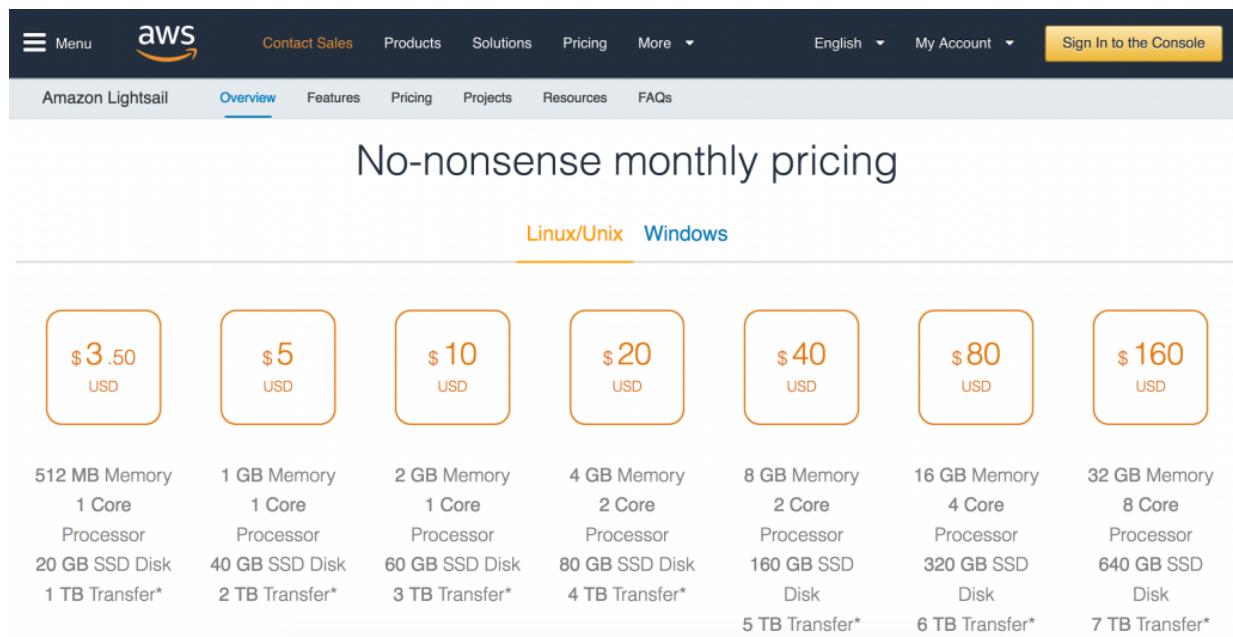


Figure 4.2: 아마존 웹 서비스 과금정책

이러한 가격정책은 현실에서도 매우 자주 목격된다. 온라인 게임의 경우 초기에 일정한 유저를 확보하기 전까지는 무료로 제공되다가 시간이 지나 유료 모델을 도입하는 전략을 사용한다. 가전제품의 경우 제품을 구입하기 전 일정한 기간동안 무료로 쓴 이후 구입할 수 있도록 하기도 한다. 아마존 AWS의 경우 낮은 성능의 인스턴스를 낮은 비용으로 제공하고 트래픽이 증가되는 과정에서 자연스럽게 고성능, 고비용의 인스턴스를 구입하도록 유도하는 경우가 대표적이다.

5장 탈중앙화 어플리케이션 블록체인

5장은 플라즈마에서 동작 가능한 다양한 토파막 플라즈마 어플리케이션과 파생되는 비즈니스 모델을 기술하고 있다. 토파막 네트워크 내에 만들어진 플라즈마 체인의 구조는 이더리움 메인넷과 동일하다. 따라서 메인넷에 이미 구동중인 DEX, 게임, 지갑과 결제등 다양한 데브(DApp)이 포팅되어 동작 가능하다. 또한 기존에 구축된 이더리움 기반의 프라이빗 체인을 플라즈마화(Plasmafy)시켜 이더리움 메인넷과 연결시키는 것도 가능하다. 이렇게 만들어진 플라즈마 체인은 다양한 용도로 활용할 수 있다. 더불어 토파막의 수수료 모델과 토큰 주조차익은 어플리케이션 이외의 분야에서 새로운 비즈니스 기회를 만들어 낼 수 있다.

다만 이하에서 설명할 비즈니스 모델은 온더(Onther Inc.)가 직접 구현하는 것이 아니며, 온더(Onther Inc.)가 아닌 참여자들의 DApp 포팅에 의하여 토파막 플라즈마에서 상정가능한 비즈니스 모델의 가능성에 대한 설명이다. 또한 이하의 비즈니스 모델의 구현에 요구되는 모든 종류의 인허가, 신고 등은 해당 비즈니스 모델을 포팅시키는 참여자들이 자신의 책임 하에 완료하여야 하는 것으로서, 온더(Onther Inc.)는 이와 관련하여 어떠한 보증이나 보장도 하지 않으며, 이와 관련하여 어떠한 책임도 부담하지 않는다.

5.1 플라즈마화된 탈중앙화 어플리케이션(Plasma DApps)

5.1.1 탈중앙화된 거래소(Decentralized exchange; DEX)

중앙화된 거래소는 사용성과 속도를 무기로 탈중앙화된 거래소에 비해 큰 점유율을 차지하고 있다. 그렇지만 2014년 마운트 곡스의 4억 5천만 달러에 달하는 암호화폐 해킹[39]을 시작으로 최근까지 수십개의 중앙화 거래소가 불분명한 이유로 사라지거나 폐업하는 일이 발생해 중앙화 거래소에 대한 문제점이 지속적으로 수면에 떠올랐다. 근본적으로 이러한 문제는 거래되는 암호자산의 프로토콜 자체는 탈중앙화 되어 있으나 이들이 거래되는 인프라 대부분은 중앙화 되어있기 때문에 발생한다.

탈중앙화 거래소는 i) 중앙서버의 개입 없이 사용자들간의 직접적인 트레이딩을 가능하게 하고 ii) 사용자들의 자금은 거래소가 아닌 사용자의 지갑에 보관하며 iii) 거래 의사표현(Order Book)과 대금 교환(Settlement) 프로세스를 블록체인 위에 스마트 컨트랙트로 구현하고 iv) 법정화폐(Fiat)나 다른 자산은 ERC20을 이용한 프록시 토큰(Proxy Token)을 사용해 블록체인에서 거래 가능한 토큰(Digix Gold, USDT 등)으로 전환해 사용하는 구조를 택한다. 이런 방식으로 Ether Delta¹, IDEX², OASIS Dex³ 같은 형태의 많은 탈중앙화 거래소가 만들어졌지만 i) 이더리움의 블록 생성 시간에 의해 생겨

¹<https://etherdelta.com/>

²<https://idex.market/>

³<https://oasisdex.com/>

나는 지연시간(Latency) ii) 거래를 오더북에 기록하고, 체결하는 등의 과정에서 발생하는 트랜잭션 수수료 등의 문제로 인한 비효율성과 좋지 않은 사용성으로 인해 시장 점유율은 높지 않다.

토카막 플라즈마 체인에서는 이미 만들어진 탈중앙화 거래소 어플리케이션을 그대로 옮겨올 수 있을 뿐만 아니라, 앞서 언급한 합의가 필요없는(No Consensus) 구조로 인해 고빈도 거래가 가능하며, 오더북에 거래를 기록하고 체결할 때마다 필요한 수수료를 일정기간 동안 사용 가능한 스테미나로 지불함으로써 사용성을 향상 시킬 수 있다.

하지만 이렇게 만들어진 고성능의 탈중앙화 거래소일지라도, 오더북의 bid와 ask를 매칭하는 것은 블록체인에 부담을 줄 뿐만 아니라 이를 단일 혹은 소수의 주체가 수행하게 될 경우 프론트 러닝(Front Running)의 문제도 발생[18][17]할 수 있다. 이 경우 탈중앙화 거래소는 오더북의 bid와 ask를 매칭 해주는 역할을 하는 마켓 메이커(Market Maker)가 붙을 수 있는 공개 인터페이스를 제공하고 이를 효율적으로 수행하는 봇을 만들어 배포하여 누구든 마켓메이커가 될 수 있도록 열어두어 경쟁토록 함으로써 bid, ask 매칭간 발생하는 프론트 러닝을 방지할 수 있다.

단, 탈중앙화된 거래소를 개설하는 경우 국제자금세탁방지기구(FATF)의 암호화폐 관련 가이드라인 및 그에 따라 한국에서도 특금법 개정이 진행 중인 점 등을 고려할 때, 개설자에게 적용될 수 있는 법령에 따른 인/허가 등의 절차는 개설자가 주의 깊게 살피고 개설자의 책임으로 필요한 인/허가를 획득해야 한다.

5.1.1.1 암호화폐 담보대출 시스템(Compound Protocol)

컴파운드 프로토콜(Compound Protocol)은 대출시장의 이자 산정 방식을 알고리즘으로 대체한 대출 플랫폼이다. 기존 P2P 대출 시스템에서 대부자(Lender)와 대여자(Borrower)의 규모가 작은 경우 쉽게 발생할 수 있는 수요/공급의 불균형을 해소하기 위해 만들어졌다.

컴파운드 프로토콜을 사용하려면 사용자 모두(대부자, 대여자)는 담보자산을 컴파운드 프로토콜의 머니마켓(Money Market) 컨트랙트에 예치(Supply) 해야 한다. 여기서 사용자가 예치(Supply)만 하고 대출(Borrow)을 하지 않는 경우⁴, 해당 자산의 폴(Pool)에서 알고리즘을 통해 산정되는 “SupplyRateIndex”에 따라 이자(금원이 아닌 암호화 화폐)를 배분 받게 된다. 대여자는 현재 담보물의 가치의 ٪에 해당하는 자산만큼을 다른 자산 폴(Pool)에서 대출 받을 수 있다. 예를들면 3 ETH를 예치하고 2 ETH에 상당하는 DAI 또는 ERC20 자산을 대출 받을 수 있다.

담보대출 방식을 살펴보면, A라는 자산을 담보로 B, C, D 등 여러 자산을 대출 받을 수 있도록 되어 있다. 하지만 이러한 담보대출 과정에서 각각의 자산의 가치를 계산하기 위한 가격 오라클이 필요하고, 지원하는 자산의 종류가 많아질수록 오라클 트랜잭션에 부담은 커진다. 토카막 네트워크에서는 다양한 자산에 대한 가격 오라클을 유지하는데 트랜잭션 수수료부담이 전혀 없기 때문에 현재 메인네트워크보다 다양한 대출 자산을 운영하는데 유리하다.

컴파운드 프로토콜은 거래소와 같은 다른 탈중앙화 어플리케이션에 탑재해 활용될 수 있다. 토카막 네트워크 내에서 만들어진 DEX에 컴파운드 방식의 대출 시스템을 붙이면, 탈중앙화된 금융거래가 가능해진다. 이러한 환경에서 사용자들은 자신의 자산을 레버리징해 거래할 수 있고, 거래소는 더욱 쉽게 유동성을 확보할 수 있는 장점이 생긴다. 이러한 대출 시스템은 토카막네트워크를 사용하는 개별 이용자가 DApp을 제작하여 선택할 수 있는 비즈니스 모델이 될 수 있다.

⁴대부자(lender)

5.1.2 크립토 키티(ERC721)

최근 ERC721의 특징인 토큰의 대체 불가능성(Non Fungible Token)을 활용해 크립토 키티와 같이 암호화 수집품(Crypto Collectible)을 만들어 사고파는 탈중앙화 게임이 많이 생겨났다. 그러나 현 시점에서 이러한 종류의 게임은 게임 내의 아이템적 요소만을 활용해 “수집품”으로서의 특징만을 강조하고 있고, 아이템을 이용한 유저간의 상호작용은 교배, 경매와 같이 매우 단순한 기능에 국한되어 있다. 유저간 상호작용을 원활하게 하기 위한 블록체인의 핵심적인 기능 요소는 i) 빠른 속도로 트랜잭션을 처리하고 ii) 트랜잭션 수행간에 발생하는 비용 부담을 어떻게 경감할 수 있느냐에 달려 있고, 토파막 플라즈마는 아이템과 유저 간의 발생하는 다양한 트랜잭션을 빠르게 처리할 수 있다.

더불어 이러한 게임들이 토파막 플라즈마에서 동작하기 위해서는 게임의 스마트 컨트랙트가 [요청가능한\(Requestable\)](#) 형태로 개발되어야 한다. 예를 들어 크립토 키티의 고양이간 교배(breeding)와 번식이 플라즈마에서 가능하게 하기 위해서는, 키티 코어 컨트랙트에서 관리하는 i) 고양이 고유값(kitty ID)을 유전정보 등을 이용해 해시화하고 ii) 소유권, iii) 고양이 데이터(유전정보 등) iv) 루트체인의 블록해시 각각을 요청 가능한 변수로 변경해야 한다. i)의 이유는 (기존)키티 코어의 개별 고양이에 대한 식별자가 1씩 커지는 자연수(auto-increment sequence)이기 때문에 플라즈마 체인으로 진입한 고양이가 교배를 한 이후 자식 고양이를 낳게 될 경우 루트체인으로 탈출하는 과정에서 고양이 식별자가 충돌(루트체인에서 번식한 고양이로 인해)할 수 있기 때문이다. ii), iii)는 조상 고양이들의 정보를 추적하기 위하여 누구나 데이터를 요청해야 하기 때문이다. iv)은 교배간 유전정보를 뒤섞는(mixing gene) 과정에서 이더리움 메인체인의 블록해시를 파라미터로 활용하기 때문이다[29].

독특한 점은 메인넷에 하나의 어플리케이션 버전이 존재할 때, 이를 수용하는 다양한 플라즈마 체인이 동시에 존재할 수 있다는 점이다. 크립토 키티가 플라즈마화 되었을 때, 키티의 고양이가 메인체인에 10,000마리의 고양이가 생성된 이후 플라즈마 체인 A에 8,000마리의 고양이가 진입(Enter)해 있는 상황에서, 플라즈마 B체인에 2,000마리의 고양이가 진입해, 같은 버전의 키티 데잍 A,B플라즈마 양 체인에 나눠서 구동되는 상황이 발생할 수 있다. 하지만 나눠진 고양이는 구조적으로만 분리되어 있을 뿐 논리적으로는 분리되어 있지 않다[11]. A플라즈마 체인의 고양이는 B플라즈마 체인으로 언제든 옮겨질 수 있고, B플라즈마 체인에서 탄생한 고양이도 언제든 A플라즈마 체인으로 이동할 수 있다. 이러한 구조가 가능한 이유는 모든 플라즈마 체인은 [요청가능 컨트랙트](#)라는 구조를 통해 탈중앙화 어플리케이션의 글로벌 상태(Global State)에 대한 논리적 일관성을 유지하기 때문이다.

플라즈마화된 크립토 게임의 설계상 가장 중요한 점은, 게임의 글로벌 상태에 영향을 주는 요소는 무엇이고 이 요소들을 어떻게 요청가능한 형태로 구현하느냐에 달려있다. 크립토 키티의 경우 가장 중요한 글로벌 상태값은 고양이의 유전정보이고, 유전정보의 생성에 가장 중요한 파라미터는 부모 고양이의 유전정보와 이더리움 메인체인의 블록해시다[29]. 요청가능한 크립토 키티 컨트랙트는 이 상태 변수들이 양 체인을 이동할 수 있도록 설계된다.

5.1.3 암호화폐 지갑과 결제

암호화폐 시장이 커져가면서 지난 몇년 간 PC와 모바일등에서 유저들이 사용 가능한 지갑(Wallet) 하드웨어 및 소프트웨어가 많아졌다. 이러한 지갑들은 키의 위치에 따라서, 장치(Device)에 직접 개인키를 저장하는 탈중앙화된 방식과 서비스 제공자가 직접 키를 보유하고 서명을 대신하는 중앙화된 두가지 방식으로 나눌 수 있다. 중앙화 지갑은 서비스 제공자가 키를 분실하거나 도난당할 위험이

있음에도 (탈중앙화된 지갑에 비해 상대적으로) 많은 인기를 끌어왔는데, 그 이유 중 하나는 중앙화된 지갑이 제공하는 편의 기능(ICO 참여, 에어드롭, 결제, 수수료 없이 전송 가능 등)과 유저가 직접 키 관리를 하는 부담이 적기 때문이다.

탈중앙화된 지갑이 중앙화된 지갑에 비해 다양한 편의기능을 제공하지 못하는 이유는, 역설적으로 키에 대한 통제 권한이 각 유저들에게 있기 때문이다. 이는 필요한 기능을 모두 메인 체인에서 처리해야 하기에 트랜잭션 수수료 및 처리속도에 대한 부담이 생기기 때문이다. 예를 들어 탈중앙화된 지갑 서비스는 유저들의 개인키 분실 부담을 줄이기 위해서 다중 서명 지갑을 유저별로 할당해 개인키 분실 시 지갑 회사와 지정된 대리인 중 2/3의 승인을 통해서 이를 복구할 수 있다. 하지만 이러한 형태의 다중 서명 컨트랙트를 메인넷에서 수십만, 수백만의 유저들에게 발급할 경우 전문학적인 트랜잭션 비용이 발생할 수 있고, 추가적으로 에스크로 스마트 컨트랙트 등 다양한 결제 옵션을 제공하게 될 경우 이 부담은 가중된다.

탈중앙화된 지갑이 토파막 네트워크상에서 구동된다면 이러한 비용은 줄어들 수 있다. 플라즈마 체인의 장점인 속도와 비용을 앞세워 다중서명 컨트랙트를 플라즈마 체인에서 만들고 이를 지갑 사용자들에게 나눠줄 수 있다. 또한 유저들에게는 스태미나 등을 통해서 일정한 기간동안 사용할 수 있는 대역폭을 할당하고 이 수준을 유저들에 따라 다양하게 조절하는 등의 방법으로 비즈니스 모델을 만드는 것도 가능하다. 또한 이때의 지갑 사용자들의 서비스 경험(UX) 또한 훨씬 개선되는데, 기존의 다른 플라즈마 지갑은 메인넷의 지갑처럼 ERC20 토큰 전송을 위해 이더리움을 꼭 보유하지 않아도 되기 때문이다.

5.1.4 탈중앙화된 가치 안정화 토큰(MakerDAO)

메이커다오의 DAI는 이더리움과 ERC20을 담보로 만들어지는 대표적인 탈중앙화된 가치 안정화 토큰이다. 메이커다오의 탈중앙화 토큰 생성 알고리즘이 동작하기 위해서는 여러 참여자들을 필요로 하며, 메이커다오 스마트 컨트랙트는 이 참여자들 각자가 스스로의 이익을 극대화하는 선택을 하는 과정에서 자연스럽게 메이커다오 생태계에 기여하도록 정교하게 설계되었다.

메이커다오 시스템은 여러 컨트랙트들로 구성되어 있고, 대표적인 것으로 가치안정화 토큰인 Dai, 토큰 생성되는 Tub, 악성 부채 청산을 위한 Tap 컨트랙트 등이 있다. 가치안정화 프로토콜은 24시간 동작하며, 각각의 컨트랙트는 지속적으로 트랜잭션을 만들어낸다. 이 중 Dai토큰 컨트랙트에서 가장 많은 트랜잭션이 발생하고 Tub, Tap순으로 이어진다. Dai, Tub, Tap의 월 평균 트랜잭션 발생량과 이 과정에서 지불되는 트랜잭션 수수료는 다음과 같다.

	월평균 Tx 건수	월평균 지불 Tx Fee	연간 사용자 부담 수수료 (1 ETH = 165.45 USD)
Dai (Sai)	15677.3	20.2 ETH	40,105.0 USD
MakerDAO (Tub, Tap)	623	1.5867 ETH	3,150.2 USD

Table 5.1: 2019.1.1 ~ 2019. 3.31 Dai, MakerDAO 트랜잭션 수수료

이 외에도 스테이블 토큰 시스템 운영을 위한 투표, Dai를 활용한 다양한 써드 파티 스마트 컨트랙트(다중서명지갑, 에스크로 컨트랙트 등)가 있으며, 생태계가 확장되는 과정에서 써드파티 컨트랙트가

만들어내는 트랜잭션은 많아지고, 수수료 부담은 커질 것이다. 유연한 수수료 정책을 도입할 수 있는 토파막 플라즈마는 이러한 요구사항을 매우 쉽게 담아낼 수 있으며, 이어지는 Native Stable Coin과 Non-Native Stable Coin는 이러한 활용 사례들을 기술하고 있다.

5.1.4.1 Native Stable Coin

토파막 네트워크에서 특정 토큰을 기반으로 담보부 증권에 해당하는 CDP⁵를 생성함으로써, 담보된 토큰 가치에 해당하는 수준의 스테이블 코인을 발생시킬 수 있다. 이는 이더리움 메인 네트워크에서도 동일하게 수행 가능하지만, 이 때의 담보 토큰의 시장 가격 정보를 기록하는(오라클) 트랜잭션이 지속적으로 일어나야 하고, 별도의 인센티브가 없다면 이러한 업무는 상당한 부담이 된다. 토파막 플라즈마에서 메이커다오의 스테이블 코인 시스템을 운영하는 경우 트랜잭션 비용을 낮춤으로써 오라클 업무에 관한 부담을 줄일 수 있다.

더불어 이렇게 만들어진 가치안정화 토큰을 요청 가능한(requestable) 형태로 구현한다면, 이 토큰을 이더리움 메인체인으로 꺼내와 이미 메인체인에서 구동중인 다양한 스마트 컨트랙트와 연동해 활용하거나, 이미 만들어진 중앙화&탈중앙화 거래소에 상장시키는 것도 가능하다.

5.1.4.2 Non-Native Stable Coin

이더리움 메인네트워크에서 Dai 토큰의 트랜잭션은 2018년 하반기 부터 증가추세에 있다. 이러한 추세라면 2020년 초에는 현재 수준대비 월평균 2.2배 정도의 트랜잭션이 발생할 것이다.

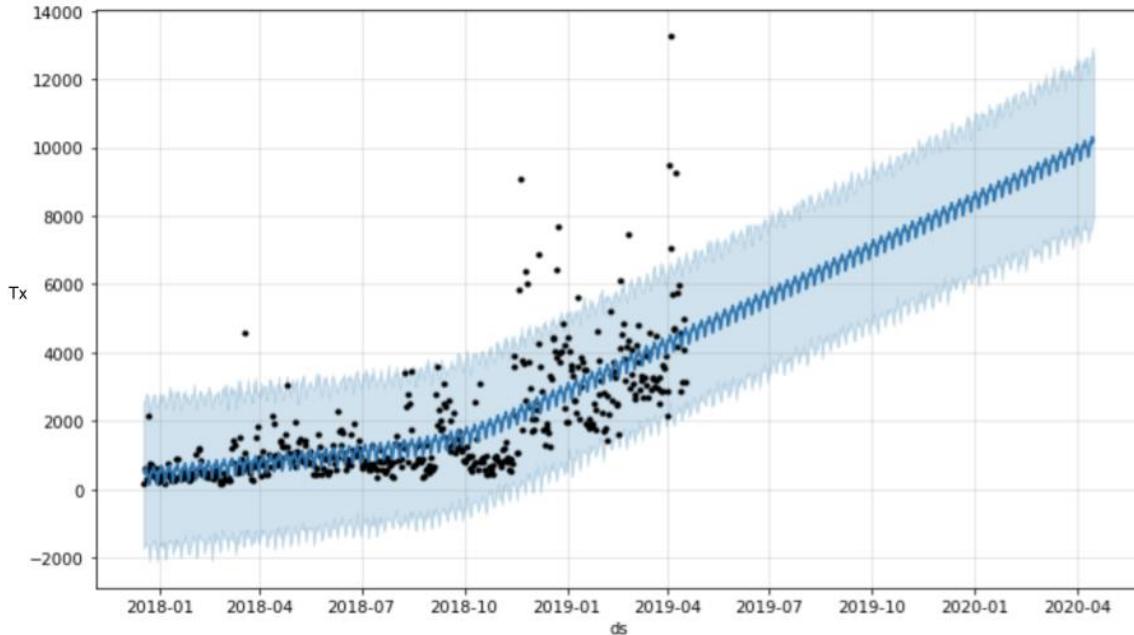


Figure 5.1: Dai 트랜잭션 발생량 및 추이

이더리움 플랫폼에 점점 더 많은 탈중앙화 어플리케이션이 늘어나는 점을 고려한다면, Dai 토큰

⁵<https://cdp.makerdao.com/>

사용을 위해 현재 더 많은 가스가격을 책정해야 하는 부담이 생길 수 밖에 없다. Dai 트랜잭션의 일부를 토파막 네트워크로 분산시켜 처리 할 수 있다면, 현재 이더리움 메인 네트워크에서 발생하는 트랜잭션 부담을 줄일 수 있다.

루트체인의 요청 가능한 토큰(Requestable Token)으로 Dai를 감싼다면 이미 발행된 Dai 토큰이 토파막 네트워크 들어오는 것(Enter)이 가능하다. 이렇게 진입한 Dai의 활용처로 가장 대표적인 것은 토파막 네트워크에서 동작하는 DEX에 기축통화로 활용되는 것이다. 뿐만 아니라 Dai를 활용한 다중 서명, 에스크로 등 복잡한 형태의 커스터디 서비스 및 결제, Dai를 담보한 스마트 컨트랙트 등 토파막 네트워크에서 Dai를 다양한 스마트 컨트랙트에 활용하는 과정에서의 발생하는 확장성 문제를 해결할 수 있다.

5.2 도메인 특정 플라즈마

5.2.1 프라이버시

5.2.1.1 익명화된 토큰(Anonymous ERC20)

영지식증명(Zero Knowledge Proof)을 비롯한 암호학 기술의 발전은 퍼블릭 블록체인 데이터의 무결성과 프라이버시를 동시에 보장할 수 있는 다양한 기법의 등장을 가속시켰다. 특히 스마트 컨트랙트의 튜링 완전성을 이용해 이더리움 프로토콜의 큰 변화 없이 탈중앙화 어플리케이션 레이어에서 토큰의 익명성을 보장할 수 있는 다양한 연구[34][2][40][1]가 이뤄졌다. 하지만 이러한 제안들은 다음의 영역에 관한 확장성 이슈로 인해 레이어1(이더리움 메인체인)보다는 레이어2(플라즈마 체인)에 더욱 적합하다.

- 노트(Note)와 같은 별도의 토큰의 소유권을 증빙하는 자료구조를 만들어 스마트 컨트랙트에 기록한다.
- 거래의 영지식 증명 검증(Verify Proof)을 스마트 컨트랙트가 수행한다.

노트(Note)란 암호화(encrypted) 및 해시화(hashed)된 토큰의 소유권으로, 각각의 노트는 i) 소유자와 ii) 액면가액(Note's Balance) 정보를 담고 있다. 거래를 위해서는 노트 소유자는 기존 노트를 소각 및 분리(split)해 새로운 노트를 만들어내는데, 만약 n개의 거래가 각각 m개의 소유권 이전을 일으킨다고 가정하면 기록해야 하는 노트 크기는 지수적(m^n)으로 증가한다. 그리고 Zether, ZETH, AZTEC, ZKDai는 이러한 노트 데이터를 관리하기 위해 스마트 컨트랙트 객체를 데이터베이스로 활용⁶하기 때문에, 스토리지 비용(Storage cost) 또한 지수적으로 증가할 수밖에 없다.

노트의 소유권이 올바르게 옮겨졌음을 증명하는 영지식 증거(ZK Proof)는 스마트 컨트랙트가 검증하는데, 검증에 필요한 가스(Gas)는 구현에 따라 최소 200만에서 700만 가스가 소모된다[2][34][1]. 이더리움 메인체인의 블록당 가스제한인 800만 가스를 기준으로 하면, 노트 전송 트랜잭션은 이더리움 메인넷 블록당 최대 4건 이상(구현에 따라 1건 이상) 담기기 어렵다.

토파막 플라즈마의 블록 가스 리밋은 3억 가스 이상[35]이기 때문에 동일한 노트 전송 트랜잭션은 블록당 30배 이상 더 담길 수 있으며, [스테미나\(Stamina\)](#)의 존재로 인해서 거래 자체에 관한 사용자

⁶ZETH는 노트를 저장하기 위해 스마트 컨트랙트로 구현된 머클트리인 MTree를 사용하고, ZKDai는 두 개의 배열을 쓴다.

들의 경제적인 비용 부담도 줄어든다. 나아가 검증을 가속하기 위한 별도의 암호 및 산술 연산자를 플라즈마EVM의 옵코드(opcode)로 탑재해 거래 확장성을 추가적으로 확보하는 것도 가능하다.

5.2.1.2 Quorum 기반 플라즈마

JP모건의 Quorum은 이더리움 구현체(go-ethereum)를 기반으로 한 프라이빗 블록체인이다. Quorum 노드의 독특한 점은 네트워크 내에서 지정된 노드들만 공유되는 상태(Private State)를 관리한다는 점이다. 만약 이 네트워크에 공개하고 싶지 않은 트랜잭션(Private Transaction)을 일으킨다면, 이 트랜잭션들은 사전에 지정된 노드 - constellation이라고 불린다 - 들간의 단방향 통신을 통해 데이터를 공유하면서 공개 수준을 조절한다[14].

Quorum 체인 또한 토파막 프로토콜을 통해 플라즈마화 될 수 있다. 이 경우 이더리움 메인 체인 컨트랙트는 퍼블릭 트랜잭션과 상태(State)의 해시를 기록하는 부분과, 프라이빗 트랜잭션과 상태 해시를 기록하는 부분으로 분리된다. 다만 이 경우 프라이빗 트랜잭션을 통해 발생한 상태 변화(state transition)에 관한 챌린지(Challenge)는 지정된 챌린저만 가능하도록 해야한다.

다만 이 경우 챌린저가 어떠한 거버넌스에 의해서 결정 되느냐가 매우 중요하다. 예를 들어 별도의 선출 거버넌스 없이 초기 오퍼레이터가 결정하는 방법도 있고, 해당 플라즈마 체인에서 사용되는 DApp 토큰의 투표를 통해서 결정하는 것도 가능하다. 토파막 네트워크는 이러한 형태의 챌린저 선출 거버넌스에는 개입하지 않는다. 다만 선출된 챌린저가 챌린지를 통해서 사전에 정의된 검증자(Verifier)로직에 의해 검증게임을 치뤄 승리하게 될 경우 오퍼레이터가 예치한 예치금은 몰수되어 챌린저에게 수여된다.

여기서 유의할 점은 검증게임이 진행되는 과정에서 트랜잭션과 상태 값의 일부가 공개될 수 있다는 점이다. 왜냐하면 검증게임은 플라즈마 체인이 아닌 이더리움 메인 체인에서 일어나기 때문이다. 이 경우 검증게임이 아닌 별도의 상태검증방법이 필요한데, 과정에서 영지식증명(Zero Knowledge Proof)은 매우 유용한 수단으로 사용될 수 있다. 영지식증명은 상태 변화(State Transition) 검증을 위해 영지식 증명과 상태 루트 해시값(State Root Hash)만을 이용해 올바른 상태 변화가 이뤄졌는지 파악할 수 있기 때문이다.

5.2.2 클라우드 저장소

이더리움 블록체인은 영구적으로 저장되는 어떠한 상태값을 기록하고 이를 수정하기 위해서 큰 비용을 지불한다. 왜냐하면 이러한 과정 자체가 상태 머클 정보를 다시 계산해야하는 부담으로 작용하기 때문이다[19]. 여기에 더해서 만약 폴노드가 되기 위해 싱크해야 할 블록과 상태 데이터의 사이즈가 커지게 될 경우 플랫폼의 탈중앙성 유지가 어려워지는 부담도 생긴다. 그렇기 때문에 큰 사이즈의 데이터를 저장하기 위해서 이더리움은 SWARM, IPFS와 같은 프로토콜을 이용하는 방향으로 발전해왔다.

하지만 순수한 IPFS⁷와 SWARM⁸ 프로토콜 자체는 인센티브 레이어가 없기 때문에 이러한 부분을 이더리움 블록체인과는 별도로 만들어야하는 부담이 있고(Filecoin⁹ 등), 블록체인 밖의 데이터를 참조할 때 생기는 오라클 이슈도 완벽하게 해결된 것은 아니다.

⁷<https://ipfs.io/>

⁸<https://swarm-guide.readthedocs.io/en/latest/introduction.html>

⁹<https://filecoin.io/>

상대적으로 큰 바이너리 데이터 저장에 부담이 적고 처리량이 큰 플라즈마 블록체인은 이에 대한 대안이 될 수 있다. 플라즈마 블록체인에 마치 데이베이스의 스키마 역할을 하는 스마트 컨트랙트를 배포하고, 이 스마트 컨트랙트가 관리하는 상태변수에 바이너리 데이터를 집어 넣으면 플라즈마 체인은 훌륭한 데이터 저장소가 된다. 이러한 형태의 저장소는 플라즈마 프로토콜을 이용하기 때문에 인센티브 레이어에 대한 고민이 필요 없을 뿐만 아니라 데이터 가용성, 데이터 위변조 위험, 오라클 문제에 대한 해결책이 될 수 있다.

5.3 파생 비즈니스

5.3.1 수수료 대출 마켓

토카막 네트워크의 플라즈마는 기존의 메인넷에서 유저들이 이더(ETH)로 매 트랜잭션마다 직접 수수료를 지불해야 하는 구조를 개선하여 [수수료 부담을 제3자가 위임](#)받을 수 있도록 만들었다. 이를 이용해 서비스 제공자들은 스스로 위임자(Delegator)가 되어서 일정한 양의 토카막 네트워크 토큰(TON)을 예치해 이를 초기 유저들에게 수수료로 제공함으로써, 유저들에게 마치 서비스에 “수수료가 없는 것처럼” 느끼게 만들 수 있다.

유저들의 수수료 부담을 줄이기 위해 서비스 제공자는 두 가지 선택을 할 수 있다. 첫번째는 서비스 제공자가 스스로 체인 오퍼레이터가 되어서 최소 가스 가격(Minimum Gas Price; MGP)을 낮추는 것이다. 두번째는 주어진 체인 내에서 다른 유저의 톤(TON)을 대출받아 일정 기간동안 수수료로 이용하는 것이다.

첫번째 최소가스가격(MGP)을 낮추는 선택을 할 경우 수수료 부담이 완전히 사라지게 되지만 수수료를 이용한 도스 공격 방지(Anti Denial of Service Attack)는 어려워진다. 따라서 적절한 수준의 안정성을 보장하면서 수수료 부담을 낮추기 위해서는 일정 기간동안 이자를 지불하고 유저들에게 나눠줄 수수료 리소스를 빌려 쓰는 것이 합리적이다.

반면 수수료 대출의 경우 다음의 두 가지 방식으로 제공될 수 있다.

1. 이더리움 메인 체인에서 주는 톤(TON, 스태미나) 대출
2. 플라즈마 체인에서 내부적으로 주는 톤(TON, 스태미나) 대출

1의 경우는 메인넷의 톤(TON) 보유자가 체인 운영자를 통해서 수수료 리소스를 대여해 주는 구조가 된다. 메인넷에 배포된 별도의 컨트렉트를 통해서 유저는 진입 예치(Enter Deposit)의 특별한 형태로 수수료를 빌려주는 목적의 톤(TON)을 예치할 수 있다. 체인 운영자는 이를 프로바이더에게 제공하고 받은 댓가를 메인넷의 톤 제공자에게 지불할 수 있다. 서비스 제공자가 직접 체인을 운영하고 있을 경우 직접 댓가를 지불할 수 있다. 필요시 이러한 거래를 자동화 할 수 있도록 체인을 여는 과정에서뱅코르(Bancor)와 유사한 형태의 유동성이 미리 확보된 DEX 프로토콜을 이용하는 것도 가능하다.

2의 경우는 마치 이오스의 chintai¹⁰, rex¹¹와 같은 서비스를 각각의 플라즈마 체인에 직접 구현하는 것이다. 유저와 유저들간에 시장논리에 의해 거래되는 수수료 마켓을 만들어 톤(TON) 차입자의 경우 많은 양의 톤(TON)을 직접 구매해 보유하지 않고도 서비스 사용량을 레버리징할 수 있고, 톤(TON)

¹⁰<https://eos.chintai.io/exchange/>

¹¹<https://eosrex.io/>

대출자의 경우 사용하지 않는 톤(TON)을 필요로 하는 유저들에게 빌려줌으로써 보유한 톤(TON)으로 이자수익을 발생시킬 수 있다.

이러한 형태의 수수료 거래소는 플라즈마 체인에 누구나 열 수 있으며, 오픈소스로 제공되는 ForkDelta¹²를 이용하거나 이미 구현된 다른 형태의 탈중앙화 거래소를 활용하는 것도 가능하다.

5.3.2 빈 블록을 제출하는 플라즈마 오퍼레이터(플라즈마 채굴 풀)

오퍼레이터는 플라즈마 블록을 루트체인에 커밋하는 과정에서 새롭게 발행되는 톤(TON)을 받는다. 이때 이렇게 발행되는 톤(TON)의 시장가치가 이더리움 메인체인에 수수료로 지불하는 이더(Ether)의 시장가치보다 클 경우 오퍼레이터는 토파막 플라즈마 체인 운영에 따른 순익이 발생한다(만약 반대의 경우 순손실이 생기게 된다).

플라즈마 운영으로만 인해서 수익이 발생하는 경우 오퍼레이터는 트랜잭션이 담기지 않은 빈 블록(Blank Block)을 만들어 루트체인 컨트랙트에 기록할 수 있다. 블록이 비는 이유는 다양하기 때문에 블록이 비어있다는 사실 자체가 플라즈마 블록을 유효하지 못한 블록으로 만들지는 못한다. 따라서 순수한 자본이득을 목적으로 체인을 운영하는 경우 채산성이 높은 시기에는 비어있는 블록을 만들어 커밋하는 오퍼레이터가 많아질 수 있다. 더불어 소액의 톤(TON)을 보유한 사용자 혹은 플라즈마 체인의 빈블록을 생성하는 과정의 복잡도가 부담이 되는 사용자의 경우 본인의 톤(TON)을 빈블록을 제출하는 오퍼레이터에 위임하여 커밋 보상을 통해 발생하는 토큰 주조차익을 나눌 수 있다. 이 때의 플라즈마 체인은 마치 작업증명(PoW)에서 해시파워를 모아서 블록보상을 나눠가지는 마이닝풀과 유사한 형태를 띠게 된다.

¹²<https://github.com/forkdelta/>

6장 기타 이슈

6.1 이더리움 2.0과 토파막 네트워크

플라즈마는 근본적으로 이더리움 메인체인의 스마트 컨트랙트를 베이스 레이어로 활용하는 레이어2 솔루션이다. 현재의 이더리움 1.0체인은 2.0체인으로 단계적인 네트워크 업그레이드[33]가 예정되어 있고, 이를 루트체인으로 활용하는 플라즈마는 (단계적인 업그레이드 과정에서) 영향을 받게 된다. 하지만 레이어1 입장에서 레이어2는 스마트 컨트랙트(탈중앙화 어플리케이션)의 일종이고, 이더리움 2.0으로의 업그레이드는 전적으로 프로토콜의 범위 내에서 일어나는 일이기 때문에 네트워크 업그레이드 과정에서 플라즈마가 받는 영향은 제한적이다[28]. 다만 플라즈마 블록의 확정(finality) 및 검증(Verification)은 각각 베이스 레이어의 확정성(finality)과 성능(performance)과 직접적인 관련이 있고, 이더리움 2.0에 활용되는 많은 기술들은 그 자체로 토파막 플라즈마의 기능을 풍부하게 한다.

6.1.1 작업증명 vs 지분증명(Proof of Work vs Proof of Stake)

현재의 이더리움(1.0)은 수정된 GHOST 프로토콜¹이라는 작업증명 알고리즘을 사용하고 있다. 포크 선택 과정에서 가장 긴(Canonical) 체인이 변경되는 것을 리오그(Reorganization)이라고 하는데 리오그가 발생하면 이미 블록에 담긴 트랜잭션들이 취소될 수도 있다. 다음 시나리오는 리오그가 발생할 수 있는 상황을 묘사하고 있다.

- 유저 A가 100단위의 토큰이 진입(Enter) 하려고 한다.
- 진입 요청을 담은 트랜잭션을 이더리움 메인체인에 전송한다.
- 해당 트랜잭션이 처리되면 이더리움 메인체인에서 100단위 토큰은 사라지고 플라즈마 체인에 100단위 토큰이 반영될 것이다.
- 이때 리오그가 발생하여 진입(Enter)에 관한 트랜잭션이 이더리움 메인체인에서 취소되면 메인체인에 사라진 100단위 토큰이 살아나고, 진입한 100단위 토큰이 플라즈마 체인에도 여전히 남아있다(이중지불 발생).

이처럼 리오그는 이중지불(double spending) 상황을 만들 수 있으며, 이러한 이유로 작업 증명은 블록의 확정성(Finality)을 100% 보장할 수 없다. 따라서, 토파막 네트워크는 리오그에 대한 대비책으로 진입/탈출 요청이 처리된 후에 바로 플라즈마 체인에 자산을 반영하지 않고 약간의 시간을 두어

¹수정된 GHOST 프로토콜은 Subtree에 포함된 블록의 개수가 많은 쪽을 메인 체인으로 선택하는 방식이다.

작업증명(PoW) 블록 높이가 충분히 쌓인 후 진입과 탈출 등의 요청(Request)을 반영하는 방법을 택한다.

반면에 이더리움2.0의 지분증명(PoS)은 Check Point를 두어 매 Check Point마다 블록을 확정(Finalize)하여 블록의 확정성(Finality)을 보장 한다[12]. 지분증명을 통해 확정된 블록은 작업증명과 달리 리오그가 발생하지 않는다. 다만 블록을 확정받기 위해서 토큰 보유자들의 지분투표를 필요로 하며, 이러한 투표기간동안 플라즈마체인에서 진입(Enter)과 탈출(Exit) 요청을 곧바로 반영할수 없다. 결론적으로 지분증명도 일정한 시간이 지난 이후에 요청(Request)를 반영해야한다. 다만 지분증명의 경우 투표가 마무리되기만 한다면 해당 블록은 100%의 확정(Finality)되기 때문에 루트체인 상황에 따라서 요청 처리에 필요한 충분한 높이를 조절해줘야하는 작업증명에 비해서 더 큰 안정성을 확보할 수 있다.

6.1.2 eWasm

eWasm은 이더리움 2.0의 로드맵에 포함되어 있는 주요 프로젝트 중 하나이며 Web Assembly를 이더리움에 이식하고자 하는 목표를 가지고 있다. Web Assembly는 C, C++, RUST 등의 상대적으로 저수준 언어(Low Level Language)를 효과적으로 컴파일 할 수 있도록 고안되었다. Wasm을 이용하면 웹에서 여러 언어로 작성된 코드가 네이티브에 가까운 속도로 실행될 수 있다.

eWasm이 적용된다면 기존 스마트 컨트랙트 실행모델인 EVM(Ethereum Virtual Machine)이 개선되면서 성능이 높아지게 된다. 또한 스마트 컨트랙트를 eWasm이 지원하는 프로그래밍 언어(C, C++, Rust, Go)로 개발할 수 있다. 또 하나의 중요한 이점은 eWasm은 독자적인 프로젝트가 아니며 Web Assembly라는 거대한 커뮤니티의 지원아래 있다는 것이다. 이러한 변화 과정에서 토탈 플라즈마도 영향을 받으며, 구체적인 내용은 다음과 같다.

6.1.2.1 토탈 플라즈마 TPS(Transaction Per Second) 개선

토탈 플라즈마는 eWasm 환경에 최적화가 가능하고, 도입되는 과정에서 TPS 개선효과는 이더리움 메인 메인체인보다 클 것으로 예상된다. 이더리움에서 eWasm이 도입되는 과정에서 성능 개선이 일어나는 과정은 다음과 같이 정리할 수 있다(Block Gas Limit은 고정, 연산 부하에 따라 유동적으로 연산 당 소모된 가스사용량을 정의한 Gas Table%²이 조정된다고 가정).

- 이더리움 체인 내에서 블록 마이닝 타임은 합의(Consensus)에 의해 결정된다.
- eWasm으로 연산 부하 감소 -> 하나의 블록에 더 많은 거래를 넣을 수 있게 된다.

이더리움 메인체인에서 블록 마이닝 타임은 합의에 의해 결정되기 때문에 연산 부하가 줄어드는 만큼 TPS가 100% 개선된다고 보장할 순 없다.

반면에 토탈 프로토콜에 eWasm이 도입된다면,

- 이더리움 체인과 마찬가지로 연산부하가 줄어들고 하나의 블록에 더 많은 거래를 넣을 수 있다.
- 토탈 플라즈마 체인에서는 합의(Consensus)가 없기 때문에 블록마이닝 타임이 합의에 의해서 결정되지 않고 순수하게 처리할 데이터량과 데이터 연산속도에 따라 결정된다.

- 따라서 처리할 데이터량이 고정되어 있다고 가정했을 때 데이터 연산 속도가 빨라진다면 블록마이닝 타임도 빨라지게 된다.

따라서 이더리움 메인체인과는 달리 플라즈마의 블록마이닝 타임은 더 확실하게 빨라지고, 이 과정에서 TPS의 개선효과는 이더리움 메인넷보다 상대적으로 더 클 것으로 예상된다.

6.1.2.2 표준 라이브러리(Standard Library)

eWasm 프로젝트의 주요 목적중 하나는 기존 언어로 스마트 컨트랙트를 작성할 수 있도록 하는 것이다. 기존 언어로 스마트 컨트랙트를 작성하는 것의 중요한 효용은 컨트랙트 개발에 필요한 툴 체인을 직접 만들 필요없이 기존 언어로 만들어진 툴 체인을 사용하면 된다는 것이다. 예를 들어 컨트랙트 개발자는 개발에 필요한 표준 라이브러리(Standard Library)를 직접 개발하지 않고 기존 언어에 있는 것을 가져와 블록체인에 라이브러리 형태로 배포할 수 있을 것이다. 다만 메인체인의 트랜잭션은 블록 Gas limit을 초과할 수 없기 때문에 만들어진 모든 표준 라이브러리(Standard Library)가 제약없이 배포될 수 있는 것은 아니다. 토카막 플라즈마 체인의 Gas Limit은 필요한 수준까지 높아질 수 있기 때문에, 컨트랙트 개발자의 라이브러리 선택의 폭은 훨씬 넓어질 것이다.

6.1.2.3 토카막 플라즈마 체인에 eWasm 적용

토카막 플라즈마는 앞으로 eWasm 도입되는 과정에서 연산 챕린지 부분을 변경해야 한다. 토카막 플라즈마 체인에서 eWASM에 연산챌린지를 적용할 때 고려해야 할 것은 EVM 컨트랙트와 eWasm 컨트랙트를 함께 사용할 수 있어야 하고, 토카막 플라즈마 체인의 기능(Request Transaction, Non-request Transaction, Escape, Challenge)에 대한 챕린지 또한 문제없이 수행할 수 있어야 한다는 것이다.

현재 토카막 프로토콜에서는 연산챌린지를 할 때 EVM의 Opcode별로 구간을 나누고 Solidity로 EVM모델을 구현한 solEVM을 실행하여 각 구간의 실행 환경 정보(Stack, Memory, Storage)등을 비교하여 검증한다. 검증값을 루트체인에 제출하기 위해 오프체인에서 검증값을 계산하기 위한 VM 가상 모델이 필요하다. 연산챌린지는 기존 컨트랙트와 eWasm 컨트랙트에 대한 검증값을 제출할 수 있어야 하기 때문에 eWasm 컨트랙트의 연산챌린지를 위해 별도로 eWasm의 Opcode 구간마다 오프체인에서 실행 환경정보값을 산출할 수 있는 solEVM과 같은 구현체가 필요하다.

6.2 플라즈마 블록체인의 확장성

플라즈마 블록체인에 유저와 트랜잭션이 늘어나면 플라즈마 체인 그 자체도 이더리움 메인넷과 같은 확장성 문제가 생길 수밖에 없다. 단일 플라즈마 체인은 단 하나의 로직으로 동작하고, 유저가 늘어나는 과정에서 단일 블록에 포함해야 하는 트랜잭션은 유저보다 더 빠르게 늘어날 수 있다. 이 경우 토카막 플라즈마의 장점인 튜링 완전성을 이용해서 플라즈마를 트리구조로 확장[32]시키는 것이 가능하다 (Figure 6.1). 레이어2 플라즈마 체인은 레이어1체인의 자식체인이다, 레이어3 플라즈마의 부모체인이 되는 것이다.

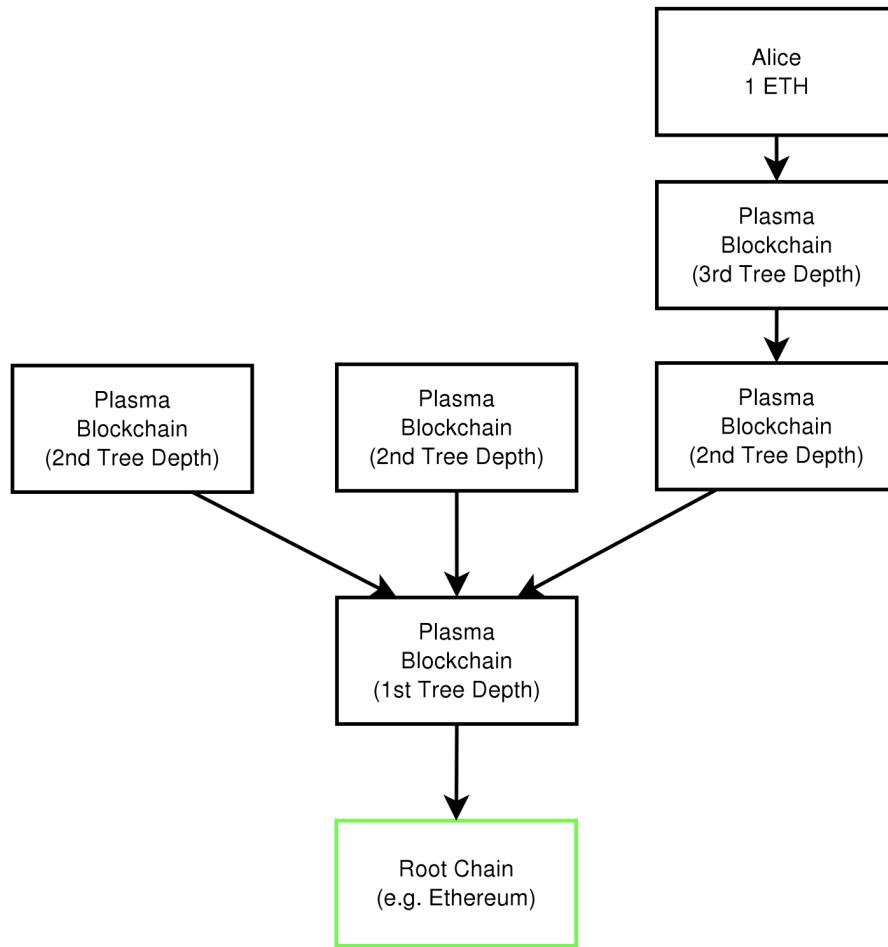


Figure 6.1: 플라즈마 트리(Plasma Tree)[출처 : 플라즈마 백서]

6.3 플라즈마 체인의 다양성

플라즈마 블록체인은 현재의 EVM을 넘어서 더 다양한 기능을 담을 수 있다. 단편적인 예로 사전에 컴파일된 컨트랙트(Pre-compiled Contract)나 추가적인 옵코드(Opcode)를 플라즈마 체인에 탑재할 수 있다. 여기서 중요한 점은 부모체인이 이러한 기능을 검증 할 수 있는 적절한 검증자(Verifier) 컨트랙트의 존재 유무와 상태 반영(State Reflection)로직이다. 만약 새로운 옵코드나 사전에 컴파일된 컨트랙트를 담았다면, 새로운 연산을 검증할 수 있는 검증자 컨트랙트를 새롭게 구현해야 한다. 또한 이 과정에서 진입 및 탈출 컨트랙트에 제약이 발생할 수 있음을 인지해야 한다. 다양한 기능을 포함한 플라즈마 체인의 예시는 다음과 같다.

- 다양한 암호 연산이 추가된 플라즈마 체인
- 다양한 대수 및 수리 통계 연산을 지원하는 플라즈마 체인
- 대안적인 상태 변화 모델을 적용한 플라즈마 체인(EVM의 대안 가상머신)

- 실험적인 수수료 모델이 도입된 플라즈마 체인

다양한 기능을 손쉽게 구현해 테스트 할 수 있는 특징으로 인해서, 토카막 플라즈마는 이더리움 로드맵이 진행되는 과정에서 새로운 기능을 테스트 할 수 있는 수단으로도 활용 가능하다.

7장 결론

토카막 네트워크는 이더리움 메인체인에 다양한 튜링 완전한 플라즈마 블록체인을 연결하고, 이더리움 블록체인의 확장성을 높이고, 사용성을 개선하는 목적을 가지고 있다. 토카막 프로토콜은 루트체인에 구현된 스마트 컨트랙트 검증자를 통해서 다양한 기능을 가진 튜링 완전한 플라즈마 체인을 연결할 수 있으며, 이를 블록체인 간 이뤄지는 데이터의 진입과 탈출에 필요한 규칙을 프로그래머가 직접 정의해 사용함으로써, 체인간 탈중앙화된(Trustless) 상태 교환이 가능하다. 이렇게 만들어진 튜링 완전한 토카막 플라즈마체인은 탈중앙화된 거래소, 게임, 지갑과 결제, 데이터 저장소, 가치안정화토큰 등 다양한 영역에서 활용될 수 있으며 블록체인 데이터의 프라이버시를 보장하는 것도 가능하다.

이 과정에서 토카막 네트워크 토큰(TON)은 토큰 주조 차익을 활용한 적절한 인센티브와 체인 예치금 및 챌린지를 통한 디스인센티브 정책에 사용된다. 이러한 경제모델은 오퍼레이터가 탈중앙성을 유지하면서 올바른 행동을 지속하는 동인으로 작용할 것이다.

토카막 프로토콜의 “올바른 상태 변화 검증”은 다양한 영역과 산업에 최적화된 플랫폼을 만드는데 매우 유용하게 활용될 수 있다. 기존의 중앙화된 방식으로 블록체인간 데이터를 교환하는 방식이나 체인간 데이터를 중개하는 별도의 블록체인 레이어를 만들어 낼 필요 없이 매우 단순하면서도 탈중앙화된 방식으로 일관된 상태 변화를 일으키는 것이 가능하다. 토카막 플라즈마는, 토큰 전송이라는 특정한 목적으로만 활용했던 전통 플라즈마의 응용 영역을 확장시켜 원래의 이더리움이 목표했던 금융과 비금융분야의 수많은 종류의 서비스를 만들 수 있는 주된 수단으로 자리잡을 것이라 확신한다.

부록

.1 용어

.1.1 공통

- 이더리움 메인 체인(Ethereum main chain) : 이더리움 메인넷, 이더리움 퍼블릭체인
- 토카막 플라즈마 체인(Tokamak Plasma chain) : 토카막 네트워크 플라즈마 체인, 토카막 프로토콜을 통해 만들어진 플라즈마 체인, 플라즈마EVM이 코어로 사용된다
- 루트체인(Root Chain) : 부모체인. 자식체인인 플라즈마 체인의 상태 변화 과정을 감시하고 평가하고 판단한다
- 플라즈마 체인(Plasma Chain) : 루트체인의 자식체인. 루트체인의 상태 변화 제약에 결여있다
- 플라즈마 오퍼레이터(Plasma operator) : 토카막 플라즈마 체인을 운영하는 주체, 플라즈마 블록 마이너(miner)
- 서비스 제공자(Service Provider) : 토카막 플라즈마 체인에서 서비스를 구동시키는 AA(Autonomous Agent), DAO(Decentralized Autonomous Organizations), DO(Decentralized Organizations), DAC(Decentralized Autonomous Corporations), DC(Decentralized Corporations)¹.
- 유저, 사용자(User) : 이더리움 계정을 가지고 있는 플라즈마 블록체인, 서비스 이용자
- 상태 변화(State Transition) : 트랜잭션으로 인해 발생하는 해당 계정의 변화
- 챌린지(Challenge) : 토카막 플라즈마 체인에서 제출된 블록이 유효하지 않은 경우 이를 바로잡기 위한 조정 과정
- 챌린저(Challenger) : 챌린지를 한 유저

.1.2 시스템 구성

- 상태 반영(State reflection) : 요청으로 인한 상태 변경, 널 계정(Null-Address)에 의해 이루어 진다

¹<https://blog.ethereum.org/2014/05/06/daos-dacs-das-and-more-an-incomplete-terminology-guide/>

- 진입(Enter) : 이더리움 메인넷에서 토파막 플라즈마 체인으로 어카운트 스토리지(암호 자산 등) 등을 이전하는 행위
- 탈출(Exit) : 토파막 플라즈마 체인에서 이더리움 메인넷으로 어카운트 스토리지(암호 자산 등) 등을 이전하는 행위
- 커밋(Commit) : 플라즈마 체인이 루트체인에 블록의 요약본을 제출하는 행위
- 검증(Verification) : 플라즈마 블록의 상태가 유효한지 판단하는 과정. 토파막 플라즈마는 트루빗 방식의 검증게임(=챌린지)를 통해 플라즈마 블록의 유효성을 판단한다
- 검증자(Verifier) : 플라즈마 상태 변화(State Transition) 과정의 유효성을 판단하는 스마트 컨트랙트.
- Continuous Rease : 플라즈마EVM에서 사용한 데이터 가용성 솔루션
- 검증게임(Verification Game) : 챌린저, 검증자, 오퍼레이터 간에 상호작용을 통해 이루어지는 검증 과정
- 요청(Request) : 플라즈마 관계에 있는 체인간 상태 반영(State Reflection)을 일으키기 위한 사용자의 요청 트랜잭션.
- 요청 트랜잭션(Request Transaction) : 진입 및 탈출을 요청하는 트랜잭션
- 요청 가능 컨트랙트(Requestable Contract) : 진입 및 탈출 인터페이스를 정의한 스마트 컨트랙트
- 요청 가능한 변수(Requestable Variable) : 진입 및 탈출 가능한 스마트 컨트랙트 변수
- 비상 탈출 요청(Escape Request) : 플라즈마 EVM에서 데이터 가용성을 보장하기 위해 높은 우선순위로 처리되는 요청 트랜잭션의 한 형태
- 루트체인 컨트랙트(Rootchain Contract) : 루트체인에 구현된 스마트 컨트랙트로 체인 예치금, 검증, 요청 등 플라즈마 체인을 관리하는 역할을 한다
- 널-계정(Null-Address) : 0x00..00 주소를 가진 이더리움 어카운트
- 최소가스가격(Minimun Gas Price) : 루트체인에 규정된, 플라즈마 체인에 포함되는 트랜잭션의 최소 가스 가격
- 블록 인질 공격(Block Withholding Attack) : 플라즈마 오퍼레이터가 플라즈마 블록의 내용을 숨기는 행위

.1.3 경제모델 및 기타

- 톤(TON) : 이더리움 메인체인에 발행된 토파막 네트워크 토큰(ERC20). 진입된 톤은 트랜잭션 수수료 및 플라즈마 체인 예치금으로 사용된다

- 체인 예치금(Chain staking) : 오퍼레이터가 토카막 플라즈마 체인을 운영하기 위해 예치한 톤(Ton) 토큰, 챌린지 당할 경우 삭감된다
- 진입 예치금(Enter staking) : 유저가 토카막 플라즈마 체인에 진입한 톤(Ton) 토큰
- 스태미나(Stamina) : 톤(Ton)으로 대납 가능한 트랜잭션 수수료의 단위, 특정 트랜잭션 송신자는 스태미너의 한도 내에서 트랜잭션을 보낼 수 있다
- 수수료 위임자(Delegator) : 토카막 플라즈마에서 수수료를 위임시킨 유저
- 수수료 수임자(Delegatee) : 수임자(Delegatee)는 자신의 스태미너 한도 내에서 위임자(Delegator)의 트랜잭션 수수료를 대신 부담한다
- 스태미나 쌍(Stamina Pair) : 수입자-위임자 관계를 형성하면 이 두 계정을 스태미나 쌍이라 칭한다
- 오라클(Oracle) : 블록체인의 외부 데이터를 블록체인 내부에 기록하는 행위
- 오라클 트랜잭션(Oracle Transaction) : 오라클 업무를 수행하는 트랜잭션

References

- [1] Arpit Agarwal. *ZkDai—Private DAI transactions on Ethereum using Zk-SNARKs*. URL: <https://medium.com/@atvanguard/zkdai-private-dai-transactions-on-ethereum-using-zk-snarks-9e3ef4676e22>. (accessed: 05.26.2019).
- [2] Benedikt Bünz et al. *Zether: Towards Privacy in a Smart Contract World*. URL: <https://crypto.stanford.edu/~buenz/papers/zether.pdf>.
- [3] Johann Barbie. *Plasma Leap - a State-Enabled Computing Model for Plasma*. URL: <https://ethresear.ch/t/plasma-leap-a-state-enabled-computing-model-for-plasma/3539>. (accessed: 04.28.2019).
- [4] Johann Barbie. *Why Smart Contracts are NOT feasible on Plasma*. URL: <https://ethresear.ch/t/why-smart-contracts-are-not-feasible-on-plasma/2598>. (accessed: 05.04.2019).
- [5] block.one. *EOS.IO Technical White Paper v2*. URL: <https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md>.
- [6] Vitalik Buterin. *A Next-Generation Smart Contract and Decentralized Application Platform*. URL: <https://github.com/ethereum/wiki/wiki/White-Paper>. (accessed: 04.28.2019).
- [7] Vitalik Buterin. *Fixed fees aren't that bad*. URL: <https://ethresear.ch/t/fixed-fees-arent-that-bad/935>. (accessed: 05.04.2019).
- [8] Vitalik Buterin. *Layer 1 Should Be Innovative in the Short Term but Less in the Long Term*. URL: https://vitalik.ca/general/2018/08/26/layer_1.html. (accessed: 04.28.2019).
- [9] Vitalik Buterin. *Minimal Viable Plasma*. URL: <https://ethresear.ch/t/minimal-viable-plasma/426>. (accessed: 04.28.2019).
- [10] Vitalik Buterin. *Plasma Cash: Plasma with much less per-user data checking*. URL: <https://ethresear.ch/t/plasma-cash-plasma-with-much-less-per-user-data-checking/1298>. (accessed: 04.28.2019).
- [11] Vitalik Buterin. *The Meaning of Decentralization*. URL: <https://medium.com/@VitalikButerin/the-meaning-of-decentralization-a0c92b76a274>. (accessed: 05.13.2019).
- [12] Vitalik Buterin and Virgil Griffith. *Casper the Friendly Finality Gadget*. URL: <https://arxiv.org/abs/1710.09437>.

- [13] Aiden Park Carl Park and Kevin Jeong. *Plasma EVM*. URL: <http://tokamak.network/kr/tech-paper.pdf>.
- [14] jp morgan chase. *quorum*. URL: <https://github.com/jpmorganchase/quorum>. (accessed: 05.14.2019).
- [15] CryptoKitties. *Herding one-million cats*. URL: <https://medium.com/cryptokitties/herding-one-million-cats-7dbec6c77476>. (accessed: 04.28.2019).
- [16] Rasmus Dahlberg, Tobias Pulls, and Roel Peeters. *Efficient Sparse Merkle Trees*. URL: <https://eprint.iacr.org/2016/683.pdf>.
- [17] Philip Daian et al. *Flash Boys 2.0 : Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges*. URL: <https://arxiv.org/pdf/1904.05234.pdf>.
- [18] Shayan Eskandari, Seyedehmahsa Moosavi, and Jeremy Clark. *Transparent Dishonesty: Front-running Attacks on Blockchain*. URL: <https://arxiv.org/pdf/1902.05164.pdf>.
- [19] *EVM: SSTORE/SLOAD gas costs split into processing and storage costs*. URL: <https://github.com/ethereum/EIPs/issues/142>. (accessed: 05.05.2019).
- [20] *Finite-state machine*. URL: https://en.wikipedia.org/wiki/Finite-state_machine.
- [21] FIO. *Blockchain Usability Report*. URL: <https://fio.foundation/wp-content/themes/fio/dist/files/blockchain-usability-report-2019.pdf>.
- [22] Danny Her et al. *Plasma EVM Economics Paper : The Mechanism Design and Economic Philosophy (v.0.9)*. URL: <https://hackmd.io/s/rJgPxWYTm#2-%EC%9C%A0%EC%A0%80%EC%9D%98-Exit%EC%97%90-%EB%8C%80%ED%95%9C-Challenge--Exit-Challenge>.
- [23] Kevin Jeong. *Plasma MVP Audit(Script)*. URL: <https://medium.com/onther-tech/plasma-mvp-audit-%EB%85%B9%EC%B7%A8%EB%A1%9D-script-166a2c5012b4>.
- [24] Kevin Jeong and Carl Park. *Plasma EVM's requestable smart contract examples*. URL: <https://github.com/Onther-Tech/requestable-contract-examples>. (accessed: 05.13.2019).
- [25] Kevin Jeong et al. *EVM Compatible Transaction Fee(GAS) Delegated Execution Architecture*. URL: <https://hackmd.io/s/SkxNKAXU7>.
- [26] Kevin Jeong et al. *EVM Compatible Transaction Fee(GAS) Delegated Execution Architecture for Plasma Chain*. URL: <https://ethresear.ch/t/evm-compatible-transaction-fee-gas-delegated-execution-architecture-for-plasma-chain/3106>.
- [27] Ben Jones and Kelvin Fichter. *More Viable Plasma*. URL: <https://ethresear.ch/t/more-viable-plasma/2160>.
- [28] Raul Jordan. *How to Scale Ethereum: Sharding Explained*. URL: <https://medium.com/prysmatic-labs/how-to-scale-ethereum-sharding-explained-ba2e283b7fce>. (accessed: 05.13.2019).
- [29] Carl Park. *CryptoKitties in Plasma EVM (KR)*. URL: <https://medium.com/onther-tech/cryptokitties-in-plasma-574159c581dc>. (accessed: 05.26.2019).

- [30] Carl Park. *Examples and Best Practices for Requestable Contracts*. URL: https://youtu.be/M650q_rbHGs. (accessed: 05.13.2019).
- [31] Learn Plasma. *Plasma Cash*. URL: <https://www.learnplasma.org/en/learn/cash.html>.
- [32] Joseph Poon and Vitalik Buterin. *Plasma: Scalable Autonomous Smart Contracts*. URL: <https://plasma.io/>. (accessed: 04.28.2019).
- [33] James Ray. *Sharding roadmap*. URL: <https://github.com/ethereum/wiki/wiki/Sharding-roadmap>. (accessed: 05.13.2019).
- [34] Antoine Rondelet and Michal Zajac. *ZETH: On Integrating Zerocash on Ethereum*. URL: <https://arxiv.org/abs/1904.00905>.
- [35] Thomas Shin. *Plasma EVM Performance Test*. URL: <https://medium.com/onther-tech/plasma-evm-%EC%84%B1%EB%8A%A5-%ED%85%8C%EC%8A%A4%ED%8A%B8-ff3a66c7fdaf>. (accessed: 05.26.2019).
- [36] Steem : An incentivized, blockchain-based, public content platform. URL: <https://steem.com/steem-whitepaper.pdf>.
- [37] Jason Teutsch and Christian Reitwießner. A scalable verification solution for blockchains. URL: <https://people.cs.uchicago.edu/~teutsch/papers/truebit.pdf>.
- [38] Toshi Times. EOS Stumbles Once More as Speculation Causes RAM Prices to Spike Beyond Affordability. URL: <https://toshitimes.com/eos-stumbles-once-more-as-speculation-causes-ram-prices-to-spike-beyond-affordability/>. (accessed: 05.03.2019).
- [39] wikipedia. Mt.Gox. URL: https://en.wikipedia.org/wiki/Mt._Gox. (accessed: 05.14.2019).
- [40] Dr Zachary J. Williamson. The AZTEC Protocol. URL: <https://github.com/AztecProtocol/AZTEC/blob/master/AZTEC.pdf>.
- [41] Gavin Wood. ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER. URL: <https://ethereum.github.io/yellowpaper/paper.pdf>.