Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.29, NO.6, Dec. 2019 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2019.29.6.1413

단방향 해시 함수를 활용한 효율적인 Off-chain Payment Channel 구현 및 실험*

김 선 형,† 정 재 열, 정 익 래‡
고려대학교 정보보호대학원 정보보호학과

Implement and Experiment of Efficient Off-Chain*

Sun Hyoung Kim,† Jae Yeol Jeong, Ik Rae Jeong‡
Division of information security, Korea University

요 약

암호 화폐는 블록체인 합의 프로토콜의 확장성 문제들로 인하여 실제 지불 수단으로 사용되기에는 한계점이 존재하였으며, 이러한 한계점들을 해결하기 위한 다양한 오프체인 솔루션들이 연구되고 있다. 본 논문에서는 단방향 해시 함수를 활용한 효율적인 오프체인 결제 채널을 설계하고 설계한 결제 채널을 이더리움 스마트 컨트랙트를 사용하여 구현하였으며 이전에 구현되었던 플라즈마 MVP와 동일한 환경에서 배포하여 각 메소드에 대한 실행 시간과 비용을 측정 및 분석하는 실험을 진행하였다. 그 결과, 플라즈마 MVP와 비교하여 제안된 솔루션은 전체 누적 시간의 경우에는 약 34%로 단축할 수 있었으며 전체 실행 비용은 약 41%로 절약할 수 있었다.

ABSTRACT

Cryptocurrency has limitations to be used as an actual payment method due to the scalability problem of the blockchain consensus protocol, and various off-chain solutions to solve these limitations are being studied. In this paper, we design an efficient off-chain payment channel using one-way hash function and implement the designed payment channel using Ethereum smart contract. In addition, the experiment was conducted to measure and analyze execution time and cost for each method by deploying it in the same environment as the previously implemented plasma MVP. As a result, compared with plasma MVP, the proposed solution was able to reduce the total cumulative time by about 34% and reduce the overall execution cost by about 41%.

Keywords: Blockchain, Payment Channel, Hash chain, Scalability

I. 서 론

최근 블록체인(Blockchain)기술은 데이터의 공개성과 투명성을 보장하며 데이터 변조로부터 무결성을 유지할 수 있다는 장점 때문에 기존의 다양한 솔루션에 사용될 수 있는 방법에 대해 활발하게 논의되고 있다[1-3].

하지만, 대부분의 블록체인 합의 프로토콜에는 확장성의 한계가 존재한다[4]. 확장성 문제는 다음과 같이 3가지로 분류할 수 있다.

처리량(Throughput): 현재 많은 블록체인은 블록 크기 제한으로 인해 트랜잭션이 블록에 포함되기를 기다리는 문제가 있다. 또한, 블록 생성시간이 짧을수록 더 많은 포크가 발생할 수 있으므로 블록 생

Received(11. 05. 2019), Modified(12. 02. 2019), Accepted(12. 03. 2019)
* 본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업의 수행결과로 추진되었음(2015-0-00936)
† 주저자, kimshzxc@korea.ac.kr
‡ 교신저자, irjeong@korea.ac.kr(Corresponding author)

성 시간을 매우 작게 설정할 수 없다. 이러한 이유로 인하여 처리량이 제한된다.

비용(Cost): 거래가 생성되고 나면, 사용자는 채굴자에게 거래 수수료를 지불해야 하며, 이는 소액 결제와 같은 가벼운 거래에 대해서도 청구된다. 따라서, 현재 블록체인에서 이러한 거래를 처리하는 것은 비용적인 문제로 제한될 수 있다.

용량(Capacity): 모든 트랜잭션이 체인에 저장될 경우 체인의 용량이 너무 커져서 체인을 유지 관리하는데 어려움이 발생한다. 2019년 9월 기준으로, 비트코인(Bitcoin) 및 이더리움(Ethereum)의 블록체인 크기는 각각 237.08GB[5], 427.06GB[6]이다.

이러한 문제를 해결하기 위하여 블록체인 외부에서 트랜잭션을 처리하여 확장성을 높일 수 있도록 다양한 오프체인 솔루션(Off-chain solution)들이 제안 및 개발되고 있다. 이들은 Fig. 1과 같은 구조로 구성되어 있으며, 메인 체인의 상태를 유지하고 다른 채널에서 처리된 마지막 상태를 적용하기 때문에 상태 채널 솔루션(State-channel solution)이라고도 불린다.

즉, 두 사용자는 서로 거래할 금액을 메인 체인의 거래에 담아 서로의 합의 없이는 사용할 수 없는 상태로 등록해둔 상태로 거래를 시작한다. 이후, 오프체인의 결제 채널을 통하여 이 거래를 해제할 수 있는 합의된 메시지 또는 거래에 거래할 금액을 담아 서로 교환하며 거래를 진행한다. 마지막으로, 거래가 완료된 경우 가장 마지막 거래를 메인 체인에 등록하여 메인 체인에 등록하였던 두 사용자의 자산을 거래 정산 금액에 따라 분배하여 거래를 종료한다.

이러한 오프체인 솔루션의 구조는 다음과 같이 확장성 문제 3가지를 모두 해결할 수 있다. 먼저, 메인 체인이 아닌 별도의 채널을 통하여 거래를 진행하기 때문에 속도와 처리량의 제한이 없다. 또한, 가장 처음과 마지막 거래만 채굴자에게 전송하는 구조로 채널을 통하여 진행되는 중간 거래들은 메인 체인에 등록하지 않으므로 소액 결제와 같은 가벼운 거래들을 채널에서 중간 비용 없이 처리할 수 있어 비용 문제를 해결할 수 있다. 또한, 마지막 거래를 메인 체인에 등록하고 중간의 거래는 메인 체인에 포함되지 않아 메인 체인의 용량 문제를 크게 줄일 수 있다.

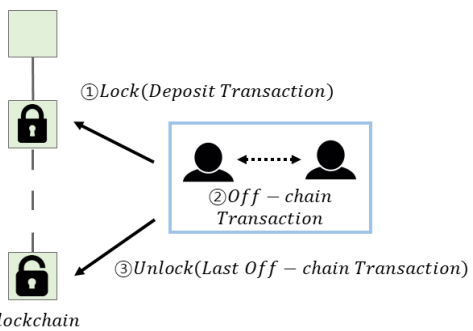

Fig. 1. An overview of existing off-chain solution

하지만, 이러한 오프체인 솔루션들의 중간 거래들은 계속하여 전자 서명과 검증을 포함하기 때문에 소액 결제를 생성하고 처리하는 IoT(Internet of Things) 또는 임베디드(Embedded) 장비와 같은 제한적인 환경의 경우 자원 소모에 대한 부담이 크다.

우리는 이 논문에서 단방향 해시 함수를 활용한 해시 체인(Hash chain) 구조를 사용하여 기존에 제안되었던 이더리움 블록체인의 오프체인 솔루션 중 하나인 플라즈마(Plasma)의 경량화된 버전으로 시간과 비용을 감소시키며 플라즈마의 기능들을 지원할 수 있는 오프체인 솔루션을 제안 및 설계하였다.

본 논문의 2장은 제안한 솔루션에 대한 배경 지식을 설명한다. 3장은 기존에 설계되어있던 플라즈마 MVP(Minimum Viable Plasma)의 메소드(Method)에 대하여 설명한다. 4장은 본 논문에서 설계한 솔루션의 메소드와 알고리즘 설명을 다룬다. 5장은 설계한 솔루션을 이더리움에서 스마트 컨트랙트(Smart contract)로 구현하고 배포하여 실제로 개발된 플라즈마 MVP 코드와 비교하여 각 메소드를 처리하는데 발생하는 비용인 가스(Gas)와 시간을 비교하여 제안된 솔루션의 효율성을 정량적으로 평가한다. 마지막 6장은 설계한 솔루션에 대한 결론과 향후 연구를 서술하였다.

II. 배경 지식

2.1 관련 연구(Related work)

비트코인[7]에서 사용자들은 비트코인에서 사용되는 스크립트(Script)로 작성된 거래를 스택 기반으로 실행하여 처리한다. 스크립트는 미리 설정해둔 작업코드(Operation code)들로 구성되어 단순한 작업코드의 조합만을 지원하며, 반복문을 지원하지 않아 튜링 불완전성(Turing incomplete)의 특성을

갖는다. 블록의 크기는 최대 1MB이며, 평균적으로 10분마다 1개의 블록을 생성하도록 1주일 간격으로 블록 난이도가 조절된다. 이러한 특성 때문에 초당 최대 7개의 거래를 처리할 수 있다.

라이트닝 네트워크(Lightning network)[8]는 비트코인에서의 확장성 문제를 해결하기 위하여 개발된 오버레이(Overlay) 네트워크 형태의 오프체인 지불 프로토콜이다. 라이트닝 채널(Lightning channel)이라는 거래 채널들로 구성된 네트워크를 라이트닝 네트워크라고 한다. 두 사용자가 자신의 자산에서 채널에 예치할 만큼의 자산을 넣어 잠그는 스크립트(Locking script)를 작성하고 해당 거래를 비트코인 메인 블록체인에 등록하여 예치(Deposit)한다. 이후, 라이트닝 채널을 통해 해당 거래를 푸는 해제 스크립트(Unlocking script)를 담은 거래를 공유하며 거래를 진행한다. 최종적으로 거래를 마무리하게 되면 가장 마지막 거래를 비트코인 메인 블록체인에 등록한다. 해당 거래의 해제 스크립트가 유효하게 되면 서로에게 거래의 적혀있는 자산만큼을 분배하게 되며 거래가 종료된다. 기본적으로 일대일로 채널을 성립시키기에 일대일 거래만 가능하도록 설계되었지만, 다른 채널들을 중계 채널로 사용하여 자신과 채널이 성립되어 있지 않은 다른 노드들과도 거래가 가능할 수 있다.

SVLP(Secure versatile light payment)[9]는 전자 서명과 단방향 해시 함수를 사용하여 오프체인 지불 프로토콜을 설계하였다. 지불자(Payer)와 수취인(Payee)은 거래 시, 단방향 해시 함수만을 사용하도록 설계하여 지불할 값에 따라서 미리 설정해둔 해시 체인의 역상(Preimage)을 제출하며 거래를 진행할 수 있다. 따라서, 라이트닝 네트워크와 비교하여 저전력 및 네트워크 자원이 부족한 상황에서도 사용될 수 있다. 하지만 설계한 메소드 중, 메인 체인으로 자금을 인출하는 $Settle()$ 과정에 대해서 채굴자들이 항상 설정된 마감 기한인 T_{ij}를 항상 상기하여 기한이 지난 경우, 자동으로 $Settle()$ 메소드가 처리된다는 가정은 블록체인에서 채굴자의 기능에 맞지 않은 설정으로 보인다.

이더리움[10]은 기존 비트코인과 다르게 튜링 완전(Turing complete) 언어를 지원하여 반복문과 조건문을 지원한다. 또한, 코딩된 규칙에 따라서 상태(State)를 변화시키는 기능이 포함된 스마트 컨트랙트(Smart contract)를 유저들이 작성하여 블록체인에 등록할 수 있다. 이러한 스마트 컨트랙트를 서버와 같이 활용하여 서버에서 실행되어야 하는 특정 작업들을 스마트 컨트랙트 내부의 메소드로 코딩하고 필요한 데이터들을 스마트 컨트랙트의 내부 변수에 저장하도록 개발하는 DApp (Decentralized Application)[11]이라는 탈중앙화된 어플리케이션을 개발할 수 있다.

플라즈마[12]는 이더리움의 오프체인 솔루션 중의 하나로 이더리움 재단에서 개발하고 있는 사이드 체인(Side chain) 솔루션이다. 이더리움 블록체인에 플라즈마 컨트랙트를 배포하여 자식 체인(Child chain)의 채널을 성립한다. 플라즈마는 현재 플라즈마 MVP, 플라즈마 Cash[13] 등 다양한 버전이 존재한다. 본 논문에서는 이더리움의 창시자인 Vitalik Buterin에 의하여 제안된 간단한 거래를 지원하는 플라즈마 MVP를 기반으로 서술한다.

2.2 사전 정의(Preliminaries)

정의. 해시 체인(Hash chain)

Leslie Lamport[16]에 의해 처음 제안된 기법으로 단방향 해시 함수의 역상 저항성을 이용하여 설계된 인증 기법이다. 증명자가 정한 임의의 값을 시드(Seed)로 이용하여 아래와 같이 검증자가 연속적으로 단방향 해시 함수를 계산하여 인증하는 방식을 사용한다.

1. 증명자(Prover)는 임의의 비밀 랜덤 값 r과 인증할 횟수 n을 선택한다.

2. 선택한 랜덤 값 r에 대하여 n번 단방향 해시 함수 f를 실행한 결과를 검증자(Verifier)에게 제출한다.

$$f^n(r) = W^0$$

3. 이후, 랜덤 값 r을 $n-1$번 단방향 해시 함수 f를 실행한 값인 W^1부터 검증자에게 제출하여 인증한다.

$$f^{n-1}(r) = W^1$$

4. 검증자는 증명자가 제출한 값에 해시 함수를 수행하여 기존에 설정된 값과 비교하여 검증한다.

$$f(W^1) = f \cdot f^{n-1}(r) = W^0$$

이러한 방식으로 r을 제출할 때까지 해시 함수의 이전 역상을 증거로 제출하여 인증하는 방법을 해시 체인이라고 한다.

또한, 전체 자산 n을 사용할 단위별로 나누어 각

단위에 따라 별도의 해시 체인을 생성하게 되면 해시 연산 횟수를 효율적으로 감소시킬 수 있어 효율적인 사용이 가능하다.

본 논문에서는 암호학적 단방향 해시 함수인 Keccak256[14]을 사용한다. 이 함수를 사용하는 이유는 이더리움에서 제공하는 해시 함수인 SHA1, SHA256과 비교하여 더 적은 가스(Gas)를 소모하기 때문이다[15].

III. 플라즈마 MVP

플라즈마 MVP는 Vitalik Buterin 의해서 제안된 사이드 체인 솔루션이다[17]. 사이드 체인이 동작하기 위해서는 블록을 생성해주는 채굴자가 필요하며, 이를 오퍼레이터(Operator)라고 한다. 이번 장에서는 플라즈마 MVP의 메소드에 대하여 서술한다.

예치(Deposit): 사용자는 사이드 체인에서 사용할 금액을 메인 체인의 플라즈마 MVP 컨트랙트로 송금한다. 오퍼레이터는 플라즈마 컨트랙트를 모니터링 하며 이더리움이 송금되어 오면 해당 사용자가 송금한 만큼의 가치를 해당 사용자의 주소에 부여하는 거래를 사이드 체인에서 생성하여 블록을 생성한다. 생성되는 블록들은 사이드 체인의 모든 사용자들에게 공개되며 해당 블록의 머클 루트(Merkle root) 값은 오퍼레이터가 플라즈마 컨트랙트에 기록된다.

지불(Pay): 현재 플라즈마 MVP의 거래들은 구

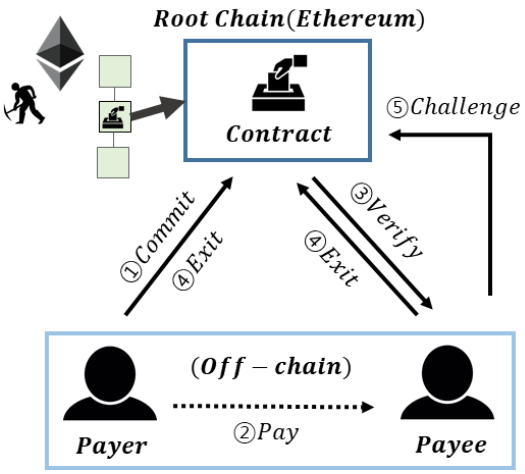

Fig. 2. An overview of proposed model

현의 편의성을 위하여 비트코인의 거래 개념과 같은 UTXO(Unspent Transaction Output) 기반으로 이루어진다. 거래에는 입력 거래 및 출력 거래를 최대 2개씩 담을 수 있다. 지불자는 자신의 UTXO 거래를 입력에 넣어 수취인의 주소와 보낼 가치를 거래의 출력에 담아 거래를 생성하고 이를 서명하여 오퍼레이터에게 전송한다. 오퍼레이터는 서명과 입력된 UTXO 거래를 검증하고 이를 모아 블록을 생성 및 전파(Broadcast)한다. 또한, 머클 루트 값을 플라즈마 컨트랙트에 기록한다. 지불자는 오퍼레이터에게 받은 블록과 플라즈마 컨트랙트의 머클 루트 값을 통해 거래의 증거가 생성되었음을 확인할 수 있다. 이후, 수취인에게 거래가 올바르게 기록되었음을 인정하는 메시지를 서명하여 전송하게 되면 수취인은 해당 거래의 가치를 사용할 수 있게 된다.

탈출(Exit): 사용자들은 자신의 UTXO 거래와 해당 거래의 머클 증거(Merkle proof)를 플라즈머 컨트랙트에 제출하여 탈출을 요청할 수 있다. 이때, 해당 거래가 해당 블록에 담겨 있는지에 대한 사실 여부는 제출된 머클 증거와 등록되어있는 머클 루트 값을 통해 검증된다. 검증된 탈출 요청은 탈출 대기열(Exit queue)에 포함된다.

도전(Challenge): 사이드 체인에서 다른 거래

Table. 1. Built-in functions for Ethereum smart contract

Function	Description
transfer	Generates a transaction so that the smart contract can transfer the number of etheres in parentheses to the address specified above.
require	If the condition in parentheses is false, it generates an error and controls the flow so that the rest of the function is not executed.
emit	This function is used to generate an event in conjunction with the Event set in the smart contract. When an event occurs through the emit function, the contents that have been set in advance in the transaction can be generated as a log and checked in the client program, which is used for interworking data between the smart contract and the client program.

의 입력 거래로 사용된 상태가 아닌지에 대한 판단은 스마트 컨트랙트에서 판단할 수 없다. 때문에, 사용자들은 이러한 악의적인 탈출 요청(Exit Request)에 대비하여 항상 플라즈마 컨트랙트에 대한 모니터링이 필요하다. 악의적인 탈출 요청이 감지되면 다른 사용자들은 해당 요청에 대하여 요청에 포함되어있는 거래가 입력값으로 사용된 거래를 증거로 제출하는 도전 프로세스(Challenge process)를 진행한다. 도전 프로세스는 정해진 보안 기간(Security period) 동안 가능하다.

또한, 채널을 운영하는 오퍼레이터가 악의적인 블록을 생성하여 인출을 시도할 수 있는 위협이 존재한다. 이러한 사항에 대해서도 모든 사용자들의 모니터링이 필요하며 악의적인 오퍼레이터의 행위가 감지되면 채널에 참가하고 있는 모든 사용자들이 자신의 모든 자산에 대하여 탈출 프로세스를 실행하여야 한다. 이를 위하여 플라즈마 컨트랙트에서는 가장 오래된 트랜잭션부터 탈출 프로세스를 진행 시킨다. 따라서, 오퍼레이터의 악의적인 탈출 프로세스 이전에 모든 사용자들의 탈출 프로세스가 먼저 진행되어 오퍼레이터의 탈출 프로세스는 이루어지지 않게 된다.

최종화(Finalize): 보안 기간 안에 유효한 도전 프로세스가 발생하지 않은 경우, 탈출을 요청한 사용자는 최종화 프로세스를 실행할 수 있다. 최종화 프로세스가 실행되면 메인 체인 상의 요청자에게 최종화 요청에 명시된 가치를 전송하는 거래를 생성하여 블록체인에 전송한다.

IV. 제안 모델

본 논문에서 제안하는 모델은 Fig.2에서 볼 수 있듯이 이더리움 기반의 오프체인 솔루션으로 오퍼레이터가 없고 경량화된 구조를 갖는 것이 장점이다. 또한, SVLP와 같이 단방향 해시 함수를 통해 거래를 진행할 수 있으며 다른 오프체인 솔루션과 비교하여 간단한 지불과 탈출 프로세스를 설계하였다. 또한, SVLP과 비교하여 마감 기한 설정이 없으며 사용자들은 언제든 자신의 예치금을 전부 또는 채널을 유지하면서 일부만 부분 인출할 수 있도록 설계하였다.

4.1 메소드

메소드에 대한 각각의 알고리즘에 대한 설명 이전에 Table 1에서 사용 중인 이더리움 스마트 컨트랙트에서 자체적으로 사용할 수 있는 함수에 대하여 설명한다.

본 논문에서 제안하는 오프체인 솔루션의 메소드는 플라즈마 MVP와 유사하게 예치, 지불, 탈출, 도전 4가지로 분류할 수 있다. 이번 장에서는 각각의 메소드와 알고리즘에 대하여 서술하였다.

예치(Deposit): 지불자 A는 수취인 B와의 거래를 위하여 비밀 값 $r_{A,B}$를 선택한 후, 자신이 채널에 예치할 총 금액인 n만큼의 단방향 해시 함수 f를 실행한 결과 값 $W^0_{A,B}$을 계산한다. 이후 이더리움 스마트 컨트랙트의 $Commit()$함수를 수취인의 주소와 $W^0_{A,B}$을 입력으로 넣고 n만큼의 이더리움을 보내 실행한다. 정상적으로 커밋이 완료되면 스마트 컨트랙트에서 해당 커밋 값을 찾을 수 있도록 인덱스(Index) 값 i를 리턴한다. 예치 프로세스의 알고리즘은 Fig.3과 같다.

지불(Pay): 예치 이후, 오프체인에서 지불자와 수취인과의 거래 프로세스는 Fig. 4와 같은 지불 및 검증 알고리즘 통하여 처리된다. 지불자 A는 비밀 값 $r_{A,B}$에 대해서 전체 자산 - 보낼 자산$(n-v)$ 만큼의 단방향 함수 실행한 결과인 $W^v_{A,B}$를 계산하여 v와 같이 전송한다. 수취인 B는 받은 자산 v가 전체 자산 n값 이하인지 확인한다. 또한, 거래로 받은 $W^v_{A,B}$ 값을 v만큼 단방향 함수 실행한 결과 값

Preset:
Payer A, Payee B.
Deposit value n, A's secret value r_{AB}
Preimage $W^0_{A,B} = f^n(r_{A,B})$
on Contract $Commit(A, B, W^0_{A,B}, n)$

Do:
$Commits[i].amount = n$
$Commits[i].payer = A$
$Commits[i].payee = B$
$Commits[i].proof = W^0_{A,B}$
$i = i+1$; // next commit index
emit i // event

Output: commit index i

Fig. 3. Deposit Algorithm

Preset:

index i

Deposit value n, value to send v

Pay:

1. Calculate Proof

$f^{n-v}(r_{A,B}) = W^{v}_{A,B}$

2. Send(i, v, $W^{v}_{A,B}$)

Verify:

$getCommit[i]$

// get information from smart contract

If $W^{0}_{A,B} = f^{v}(W^{v}_{A,B})$ and $v \leq n$ **:**

then $ACCEPT$

Else :

$REJECT$

Fig. 4. Pay Algorithm

과 컨트랙트에서 A가 커밋한 값 $W^{0}_{A,B}$을 비교하여 올바른 값일 경우 거래를 진행한다.

탈출(Exit): 탈출 프로세스는 2가지의 경우로 나눌 수 있다. 첫 번째는 지불자가 먼저 탈출을 요청하는 경우, 두 번째는 수취인이 먼저 탈출을 요청하는 경우이다. 각각의 프로세스에 대해서 경우를 나누어 설명한다.

수취인이 먼저 탈출을 요청하는 경우 Fig. 5와 같은 탈출 프로세스를 실행한다. 수취인 B는 커밋의 인덱스 값 i와 지불자로부터 받은 가치 v 및 이에 해당하는 역상 값 $W^{v}_{A,B}$을 입력 값에 넣어 스마트 컨트랙트의 $PayeeExitFirst(i, W^{v}_{A,B}, v)$함수를 실행시킨다. 해당 함수는 수취인이 입력한 역상 값 $W^{v}_{A,B}$에 대하여 입력받은 v만큼 단방향 해시 함수 f를 실행한 후, 커밋에 기록된 $W^{0}_{A,B}$ 값과 비교하여 일치 여부를 판단하여 유효성을 검증한다. 만약, 유효한 요청일 경우 수취인이 요청한 v만큼의 가치를 전송하는 트랜잭션을 메인 체인에서 발생시킨다. 또한, 수취인이 보낸 v 값이 전체 자산 n과 같을 경우 해당 커밋 정보를 삭제한다. 이때, 수취인이 보낸 v 값이 전체 자산 n보다 작을 경우, 부분 인출(Partial withdraw)로 판단하여 커밋된 내용에 대하여 전체 자산을 $n-v$, 단방향 해시 함수의 값을 $W^{v}_{A,B}$로 재설정하여 해당 채널을 유지 시킨다,

Preset:

index i

Value to exit v

Proof $W^{v}_{A,B}$

on Contract $PayeeExitFirst(i, W^{v}_{A,B}, v)$

Do:

1. Check message

require($Commits[i].payee = msg.sender$)

require($Commits[i].amount \geq v$)

2. Verify and value transfer

If $W^{0}_{A,B} = f^{v}(W^{v}_{A,B})$:

// Full exit

If $Commits[i].n = v$:

$Delete(Commits[i])$

$msg.sender.transfer(v)$

// Partial exit

ElseIf $Commits[i].amount > v$:

$Commits[i].proof = W^{v}_{A,B}$

$Commits[i].amount = n - v$

$msg.sender.transfer(v)$

Else :

$REJECT$

Fig. 5. Payee's Exit Algorithm

지불자 A가 먼저 탈출 요청을 시도할 때의 알고리즘은 Fig. 6과 같다. 지불자 A는 자신이 커밋한 인덱스 값 i와 자신이 사용한 가치 v를 제외한 나머지 잔액 R을 담아 스마트 컨트랙트의 $PayerExitFirst(i, R)$함수를 실행시킨다. 스마

Preset:

index i

Value to exit R

on Contract $PayerExitFirst(i, R)$

Do:

1. Check message

require($Commits[i].payer = msg.sender$)

require($Commits[i].amount \geq R$)

2. Put in to the exit queue

$PayerExitQueue[i].timestamp$

$= block.timestamp$

$PayerExitQueue[i].reamin = R$

Fig. 6. Payer's Exit Algorithm

트 컨트랙트는 해당 요청에 대하여 요청자가 해당 인덱스의 커밋에 기록된 지불자가 맞는지, 나머지 잔액 R 값이 전체 자산 n 이하인지 검증한 이후, 검증 통과 시에 탈출 대기열($PayerExitQueue$)의 인덱스 값 i에 나머지 잔액 R과 타임스탬프(Timestamp)를 기록한다.

최종화(Finalize): 사전에 설정해둔 안전 기간 이내에 도전 프로세스 요청이 없을 경우, 요청자는 인덱스 값 i을 담아 Fig. 7의 최종화 프로세스 $Finalize()$를 실행시킬 수 있다.

최종화 프로세스가 실행되면 해당 인덱스의 탈출 대기열을 조회하여 현재 블록의 타임스탬프와 비교하여 안전 기간이 지난 경우, 커밋을 조회하여 요청한 잔액 R 값이 커밋에 남아있는 전체 자산 n이하인지 검증한다. 검증이 완료되면 요청자에게 탈출 요청에 명시된 가치 R을 전송하는 거래를 생성하여 블록체인에 전송한다.

도전(Challenge): 만약 지불자가 부정한 인출을 위하여 자신이 소비한 값 v 보다 큰 값을 잔액 R로 설정하여 탈출 요청을 한 경우, Fig. 8의 $Challenge()$를 수행한다.

수취인은 스마트 컨트랙트의 자신이 포함된 커밋의 인덱스 값 i에 대한 탈출 프로세스가 발생했음을

Preset:

index i

Security period = $1week$

on Contract $Finalize(i)$

Do:

1. Set variables

$R = PayerExitQueue[i].remain$

$n = Commits[i].amount$

$pastTime = (block.timestamp -$

$PayerExitQueue[i].timestamp)$

2. Check message

require($Commits[i].payer = msg.sender$)

require($n \geq R$)

3. Check timestamp and value transfer

If $pastTime > 1week$:

// Full exit

If $n = R$:

$Delete(Commits[i])$

$msg.sender.transfer(R)$

// Partial exit

ElseIf $n > R$:

$Commits[i].amount = n - R$

$msg.sender.transfer(R)$

Else:

$REJECT$

Fig. 7. Finalize Algorithm

이벤트 발생을 통해 감지할 수 있다. 감지한 이벤트의 요청된 잔액 R 값이 지불자의 잔액보다 클 경우 악의적인 탈출 요청이라고 판단한다. 수취인은 자신이 지불자로부터 커밋의 인덱스 값 i, 받은 가치 v와 이에 해당하는 역상 값 $W^v_{A,B}$을 증거로 입력값에 넣어 도전 프로세스를 실행시킨다. 스마트 컨트랙트는 인덱스 값 i의 커밋 정보를 기반으로 도전 프로

Preset:

index i

Received value v, Proof $W^v_{A,B}$

on Contract $Challenge(i,v,W^v_{A,B})$

Do:

1. Check message

require($Commits[i].payee = msg.sender$)

require($commits[i].amount \geq v$)

2. Verify and value transfer

If $W^0_{A,B} = f_1 \cdots f_v(W^v_{A,B})$:

// Full exit

If $Commits[i].n = v$:

$Delete(Commits[i])$

$msg.sender.transfer(v)$

// Partial exit

ElseIf $Commits[i].amount > v$:

$Commits[i].proof = W^v_{A,B}$

$Commits[i].amount = n - v$

$msg.sender.transfer(v)$

Else :

$REJECT$

Fig. 8. Challenge Algorithm

세스 신청자가 커밋 정보의 수취인이 맞는지, 요청한 가치 v에 대하여 $n-v$가 탈출 대기열의 R보다 크고 $W^v_{A,B}$를 v번 단방향 해시 함수 수행 결과가 커밋에 기록된 $W^0_{A,B}$와 같으면 유효한 도전으로 인정한다. 유효한 도전일 경우 수취인이 입력한 v만큼의 가치를 수취인에게 전송한다.

V. 비교 및 실험

5.1 설계적 장단점 분석

설계한 오프체인 솔루션과 기존의 플라즈마 MVP와 SVLP를 비교하여 장단점을 분석한 결과를 정리한 결과는 Table 2와 같다. 먼저 기존 SVLP에서의 탈출 프로스세스에서 사용되는 마감 기한 T_{ij}을 삭제하여 채굴자들이 해당 시간을 계속 상기한 상태로 유지된다는 가정을 삭제하였다.

지불 구조에 대해서는 단방향 해시 함수를 사용하기 때문에 SVLP와 같이 1:1 단방향 지불만 가능하도록 설계하였다. 반면 플라즈마 MVP에서는 사이드 체인에 자금을 이체하면 사이드 체인에 존재하는 모든 사용자와 거래를 진행할 수 있다. 이는 제안한 구조의 단점으로 보이나, 본 논문에서 제안한 솔루션에서도 플라즈마의 Operator 같은 신뢰할 수 있는 제 3자를 중간자 역할로 등록하여 사용하면 다른 사용자들과 1:N 거래를 지원하도록 설정할 수 있다. 하지만, 플라즈마 MVP에서와 같이 Operator와 같은 신뢰할 수 있는 제 3자를 설정할 경우, 단일 실패 지점(Single point failure)이 될 수 있어 단일 실패 지점 없는 1:1 거래로 설계하였다.

플라즈마 MVP에서의 단점 중 하나인 복잡한 탈출 프로세스를 제안된 솔루션은 단방향 해시 함수의 역상에 대한 검증 절차를 통하여 단순화 하였으며 수취인의 탈출 요청의 경우 보안 기간 동안 보류되는 과정 없이 바로 처리하도록 설계하여 더 효율적인 탈출 프로세스를 구현하였다.

Table 2. Feature comparison table with existing model

Item	Proposed Model	Plasma MVP	SVLP
Payment	1:1	1:N	1:1
Critical Assume	X	X	remind T_{ij}
Single Point Failure	X	Operator	X
Exit Process	Simple	Complex	Simple
Blockchain	Ethereum	Ethereum	Bitcoin
Partial Withdraw	O	X	X

또한, 탈출 프로세스에서 본 논문에서는 기존 모델들에는 존재하지 않았던 부분 인출이라는 개념을 설계하여 채널을 유지하면서 일부의 자산만 채널에서 메인 체인으로 가져와서 사용할 수 있다는 점이 장점이다.

부분 인출에 대하여 기존에 제안된 솔루션에서 제안된 메소드를 조합하여 구현할 수 있지만 본 논문에서 제안하고 있는 솔루션과 비교하여 추가적인 과정과 이에 대한 추가적인 비용이 발생하므로 제안된 솔루션의 부분 인출 기능이 가장 우수하였다.

Table 3. Time(ms) measurement results.

Process	Proposed Model	Plasma MVP
Deposit	95	69
Exit	Payer: 68 Payee: 75	139
Finalize	77	102
Challenge	58	571
Total	305	881

5.2 구현 및 실험

이 장에서는 플라즈마 MVP 컨트랙트와 본 논문에서 제안한 오프체인 솔루션을 각각 로컬 이더리움 네트워크에 배포하여 실험한 결과를 설명한다.

이더리움 RPC 클라이언트로 Ganache CLI (v6.6.0, ganache-core v2.7.0), 컨트랙트 컴파일 및 배포를 위하여 Truffle(v5.0.34), 컨트랙트에 사용한 언어로는 Solidity (v0.5.0)을 사용하였으며 테스트 케이스 작성과 실행을 위하여 Node.js (v8.9.4), Web3.js (v1.2.1)을 사용하였다.

또한, Intel(R) Core i5-6600 3.30GHz의 CPU, 8 GB의 RAM, Windows 10의 운영체제로 구성된 PC에서 실험을 진행하였다. 실험에서 2명의 사용자를 각각 지불자와 수취인으로 설정하였다. 지불자는 10만큼의 이더리움을 컨트랙트에 전송하고 5만큼의 가치를 수취인에게 전달한다. 이후, 지불자와 수취인이 각각 5만큼의 자산으로 탈출 프로세스를 실행하는 경우와 지불자가 6의 자산으로 탈출 프로세스를 실행하여 수취인이 해당 탈출 요청에 대한 도전 프로세스를 진행하는 경우를 각각 실험하여 측정한 결과 시간과 비용의 측정 결과는 각각 Table 3, Table 4와 같았다.

예치 프로세스에서 플라즈마 MVP의 경우 예치할 자산을 담아 예치 프로세스를 실행시키면 커밋 정보 배열에 해당 주소와 예치금을 포함시키는 구조이다. 제안한 모델의 경우에는 플라즈마 MVP에서 작동하는 프로세스와 별개로 오프체인 상에서 예치할 자산만큼의 단방향 해시 함수의 역상을 계산하는 작업이 필요하여 추가적인 시간과 가스가 소모되었다.

탈출 프로세스의 경우, 플라즈마 MVP는 탈출 요청한 거래를 탈출 대기열에 추가한다. 이러한 과정에서 입력한 거래의 머클 증거, 전자 서명 그리고 UTXO의 입력값에 대한 검증을 진행하는 작업이 실행된다. 이는 비교적 비용과 시간이 많이 소모되는 연산과정이 필요한 작업이다. 제안된 모델의 경우에는 지불자와 수취인의 탈출 프로세스가 다르기 때문에 실행 시간과 비용이 각각 다르게 측정된다. 지불자가 탈출 프로세스를 신청한 경우, 탈출 대기열에 타임스탬프와 요청한 잔액을 기록하는 작업을 진행한다. 반면, 수취인이 탈출 프로세스를 신청한 경우에는 수취인이 제시한 증거에 대하여 신청한 가치만큼 단방향 해시 함수를 통해 검증한 후, 검증 통과 시에 즉시 거래를 생성하는 프로세스를 진행한다. 결과적으로 플라즈마 MVP와 비교하여 지불자의 경우 약 49%의 시간과 41%의 비용만을 소모하였으며 수취인의 경우 약 54%의 시간과 18%의 비용만 소모하였다.

플라즈마 MVP의 최종화 프로세스의 경우에는 기본적인 시간 비교와 거래 생성 프로세스 이외에 우선순위 대기열을 사용하여 가장 오래된 트랜잭션을 먼저 처리하는 프로세스를 진행한다. 반면, 제안된 오프체인 솔루션의 최종화 프로세스는 지불자의 경우에만 실행이 요구된다. 최종화 프로세스가 요청되면 탈

Table 4. Cost(gas) measurement results.

Process	Proposed Model	Plasma MVP
Deploy	2227898	5559378
Deposit	116294	111193
Exit	Payee: 47183 Payer: 106831	257101
Finalize	33119	70947
Challenge	46955	130752
Total	2531097	6129371

출 대기열에 기록된 타임스탬프와 현재 블록의 타임스탬프를 비교하여 보안 기간 이상 경과 된 경우에 거래를 생성하는 프로세스를 진행한다. 이러한 과정은 플라즈마 MVP와 비교하여 간단한 프로세스 과정이기 때문에 시간의 경우 약 25%, 비용의 경우 약 54% 절약한 결과를 얻을 수 있었다.

도전 프로세스의 경우, 플라즈마 MVP의 경우 탈출 대기열의 거래가 입력값으로 포함된 거래를 제시하여 도전 프로세스를 실행시킨다. 이때, 도전에 사용된 거래의 입력 출력 값, 전자 서명과 머클 증거에 대한 검증 프로세스를 진행하는 과정을 진행한다. 도전 프로세스의 경우 제안된 솔루션에서는 가치에 따라서 단방향 해시 함수에 대한 증거 검증 작업 이후, 거래 생성 프로세스로 진행된다. 결과적으로 제안된 솔루션의 도전 프로세스는 플라즈마 MVP의 10%의 시간, 36%의 비용만을 소모하였다.

각각의 모델에서 모든 프로세스에 대한 누적 시간과 누적 비용 각각 Fig.9와 Fig.10과 같다. 전체 누적 시간을 비교하였을 때, 플라즈마 MVP의 경우 총 881ms가 소요되었다. 하지만, 제안된 솔루션을 사용할 경우 총 305ms가 소요되어 약 66% 정도의 시간을 단축할 수 있었다. 전체 비용을 비교하였을 때, 플라즈마 MVP의 경우에는 약 613만 가스가 소모되었다. 하지만 제안된 솔루션을 사용할 경우 253만 가스가 소모되어 약 59%의 가스를 절약할 수 있었다.

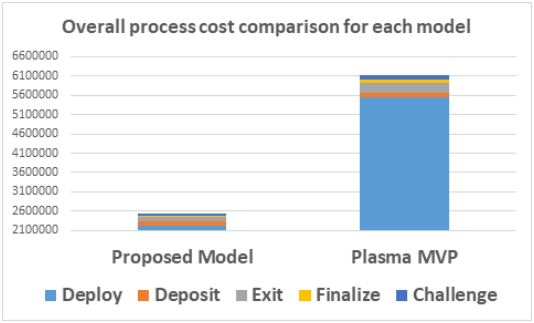

Fig. 9. Overall process cost(gas) comparison for each model

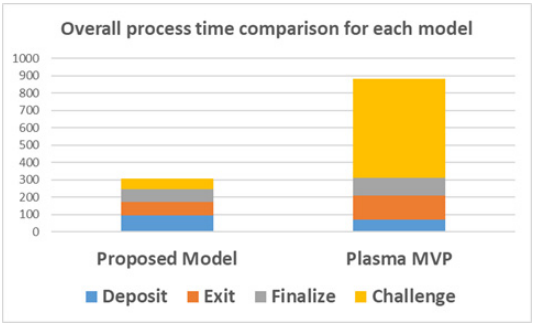

Fig. 10. Overall process time(ms) comparison for each model

VI. 결론 및 향후 연구

본 논문에서는 단방향 해시 함수를 활용한 해시 체인 구조를 사용한 경량화된 오프체인 솔루션을 제안 및 구현하였다. 제안된 솔루션은 기존에 제안되었던 플라즈마 MVP의 메소드를 경량화하여 해시 체인의 역상을 제출하는 검증 과정을 통해 구현하였다. 이후, 제안된 솔루션을 이더리움 스마트 컨트랙트로 구현하여 기존에 구현되어 있었던 플라즈마 MVP 스마트 컨트랙트와 동일한 환경에서 배포하여 각 메소드별로 실행 시간 및 실행 비용을 측정하였다. 그 결과, 예치 프로세스를 제외한 모든 프로세스에서 프로세스 실행에 필요한 시간과 비용이 감소하였다. 특히, 사용자의 악의적인 탈출 프로세스에 대응하기 위한 도전 프로세스의 경우 실행 시간이 10%, 실행 비용이 약 36%로 크게 감소하였다. 또한, 수취인의 경우 탈출 프로세스를 통하여 안전 기간 동안 대기할 필요 없이 바로 탈출할 수 있었다.

채널을 유지하면서 일부의 자산만 채널에서 메인 체인으로 가져와서 사용할 수 있는 부분 인출이라는 개념을 기존 탈출 메소드에 포함하여 설계 및 구현하였다. 부분 인출에 대하여 기존에 제안된 플라즈마 MVP와 SVLP에서 자체적인 메소드를 조합하여 부분 인출을 구현할 수 있었지만 본 논문과 같이 처음부터 이러한 기능을 포함한 것이 아니므로 때문에 추가적인 과정과 비용이 필요하였다. 따라서, 본 논문의 부분 인출 메소드가 가장 효율적인 부분 인출 메소드 임을 확인할 수 있었다.

향후 연구로는 제안한 솔루션을 통하여 스마트 컨트랙트에 설정된 수취인과 지불인 이외에 다른 채널들을 중계 채널로 사용하여 1:N 거래를 지원하는 스킴을 설계 및 구현하여 성능을 평가할 것이다. 또한, 기존에 퍼블릭 블록체인의 중요한 문제로 제기되었던 프라이버시 문제를 설계한 오프체인 솔루션으로 해결하는 연구를 진행할 예정이다.

References

[1] Dorri, Ali, et al. "Blockchain: A distributed solution to automotive security and privacy," IEEE Communications Magazine, vol. 55, no. 12, pp. 119-125, Dec. 2017.

[2] A. Stanciu, "Blockchain Based Distributed Control System for Edge Computing," 2017 21st International Conference on Control Systems and Computer Science (CSCS), pp. 667-671, May. 2017.

[3] Liang, Xueping, et al. "Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," Proceedings of the 17th IEEE/ACM international symposium on cluster, cloud and grid computing, pp. 468-477, May. 2017.

[4] Zheng, Zibin, et al. "Blockchain challenges and opportunities: A survey," International Journal of Web and Grid Services, vol. 14, no, 4, pp. 352-375, Jun. 2018.

[5] Blockchain.com, "Bitcoin Size" https://www.blockchain.com/ko/charts/blocks-size, Last accessed 30 September 2019.

[6] BitInfoCharts, "Ethereum blockchain size" https://bitinfocharts.com/ ethereum, Last accessed 30 September 2019.

[7] Nakamoto, Satoshi., "Bitcoin: A peer-to-peer electronic cash system," https://bitcoin.org/bitcoin.pdf, Oct. 2008.

[8] Poon, Joseph, and Thaddeus Dryja. "The Bitcoin Lightning Network: Scalable Off-chain Instant Payments," http://lightning.network/lightning-network-paper.pdf, Jan. 2016.

[9] Zhong, Lin, et al. "A secure versatile light payment system based on blockchain," Future Generation Computer Systems, vol. 93, no. 1, pp. 327-337, Apr. 2019.

[10] Buterin, Vitalik. "Ethereum white paper," https://github.com/ethereum/wiki/wiki/White-Paper, 2013.

[11] Cai, Wei, et al. "Decentralized applications: The blockchain- empowered software system," IEEE Access, vol. 6, no. 1, pp. 53019-53033, Sep. 2018.

[12] Poon, Joseph, and Vitalik Buterin. "Plasma: Scalable autonomous smart contracts," https://plasma.io/plasma.pdf, August. 2017.

[13] Konstantopoulos, Georgios. "Plasma Cash: Towards more efficient Plasma constructions," arXiv preprint arXiv: 1911.12095, Nov. 2019.

[14] Bertoni, Guido, et al. "Keccak sponge function family main document," Submission to NIST (Round 2), Mar. 2009.

[15] Wood, Gavin. "Ethereum: A secure decentralised generalised transaction ledger," Ethereum project yellow paper, Apr. 2014.

[16] Lamport, Leslie. "Password authentication with insecure communication." Communications of the ACM, vol. 24, no. 11, pp. 770-772, Nov. 1981.

[17] Buterin, Vitalik. "Minimal Viable Plasma," https://ethresear.ch/t/minimal-viable-plasma/426, Jan. 2018.

〈저자소개〉

김 선 형 (Sun Hyoung Kim) 학생회원
2018년 2월: 전북대학교 컴퓨터공학과 졸업
2018년 3월~현재: 고려대학교 정보보호대학원 정보보호학과 석사과정
〈관심분야〉 정보보호, 블록체인, 보안 프로토콜

정 재 열 (Jae Yeol Jeong) 학생회원
2010년 2월: 고려대학교 수학과 졸업
2013년 8월: 고려대학교 정보보호대학원 정보보호학과 석사
2013년 9월~현재: 고려대학교 정보보호대학원 박사과정
〈관심분야〉 암호 프로토콜, 프라이버시 향상 기술, 생체인증

정 익 래 (Ik Rae Jeong) 종신회원
1998년 2월: 고려대학교 전산학과 졸업
2000년 2월: 고려대학교 정보보호학과 석사
2004년 8월: 고려대학교 정보보호학과 박사
2008년 3월~현재 고려대학교정보보호대학원조교수 부교수 교수
〈관심분야〉 프라이버시 향상 기술, 데이터베이스 보안, 생체인증