스마트 컨트랙트를 이용한 분산형 토큰 교환 시스템

이전제*, 조만재*, 조석헌**
광운대학교 컴퓨터공학과*
Qualcomm Institute, University of California, San Diego**
e-mail: whssodi@gmail.com

Decentralized Token Exchange System based on Smart Contract

Jeon-Je Lee*, Man-Jae Cho*, Seokheon Cho**
Dept of Computer Engineering, Kwang-woon University*
Qualcomm Institute, University of California, San Diego**

요 약

사용자가 보유하고 있는 토큰은 기존 중앙화된 암호 화폐 거래소를 통해 교환 할 수 있다. 그러나 암호 화폐 거래소를 이용한 토큰 교환은 높은 수수료, 암호 화폐 거래소 해킹 가능성, 해당 거래소 내 등록된 암호 화폐에 대해서만 교환이 가능하다는 문제점이 존재한다. 이더리움 플랫폼에 배포된 스마트 컨트랙트는 블록체인 기반으로 다양한 형태의 계약을 조건이 만족할 시 자동으로 이행한다. 본 연구는 스마트 컨트랙트를 이용하여 기존의 중앙화된 암호 화폐 거래소 기반 토큰 교환 문제점을 해결하면서 이더리움 지갑 내 토큰을 사용자가 원하는 ERC-20 토큰으로 자동 교환해 주는 분산형 토큰 교환 시스템을 제안한다.

1. 서론

2008년 비트코인을 시작으로 많은 종류의 암호 화폐가 생겨났다 [1]. 특히, 이더리움은 2019년 기준 비트코인 다음으로 시가총액이 큰 암호 화폐이다 [2, 3]. 또한, 메인 네트워크를 가지고 있는 이더리움에는 다양한 목적을 위해 수많은 주체가 발행한 20만 개가 넘는 ERC-20 (Ethereum Request for Comment 20) 호환 토큰들이 존재한다 [4]. ERC-20은 이더리움 블록체인 네트워크에서 발행되는 토큰의 표준을 의미한다.

개별적 토큰마다 특정 목적을 위해 발행되었기 때문에, 다른 목적의 서비스를 이용하기 위해서는 토큰을 새로 구매하거나 기존의 토큰과 교환해야 한다. 예를 들어, 새로운 서비스를 이용하고 싶으면 사용자가 보유한 사용하지 않는 토큰을 이용하고 싶은 서비스에 대한 토큰으로 교환하여 사용 할 수 있다. 이처럼, 비용적 측면에서 효율적으로 토큰을 사용하기 위해서는 토큰 교환 시스템이 필연적으로 요구된다.

'Coinrail', 'Bithumb', 'Youbit' 등과 같은 기존의 중앙화된 암호 화폐 거래소를 통해서 사용자가 거래소 내에 보유한 암호 화폐를 원화 또는 달러로 환전한 뒤 원하는 암호 화폐를 사는 방식을 통해 토큰을 교환 할 수 있다. 하지만, 이와 같은 토큰 교환 방법은 높은 수수료를 암호 화폐 거래소에 지급해야 하며, 토큰에서 토큰으로 환전하지 못해 원화 또는 달러로 교환을 거쳐야 하는 번거로움이 존재한다. 또한, 중앙화된 암호 화폐 거래소는 널리 알려지지 않은 토큰들을 교환하는 데 어려움이 있고, 중앙화된 암호 화폐 거래소가 웹 서버 환경임을 이용한 APT 공격 및 웹 해킹 공격, 사용자 편의성을 위해 사용자 암호 화폐 지갑의 개인 키를 저장한 데이터베이스 해킹 공격, 암호 화폐 거래소 직원 대상의 사회 공학적 해킹 공격 등 보안 측면에서 문제를 갖는다 [5].

이러한 문제를 해결하기 위해 본 논문에서는 이더리움 블록체인 네트워크에 배포된 스마트 컨트랙트 (Smart Contract)를 이용한 분산형 토큰 교환 시스템을 제안한다.

스마트 컨트랙트는 외부 소유 계정(External Owned Account, EOA)에서 메시지를 전달하기 위해 생성하고, 서명한 트랜잭션을 받아 특정 조건을 확인한다. 특정 조건을 만족할 시 자동으로 계약을 이행하며 스마트 컨트랙트를 통해 실행되는 모든 정보는 노드에 기록되어 블록체인 네트워크에 전파되기 때문에 중앙 서버가 필요하지 않다. 또한, 블록체인의 구성요소인 노드를 이용하여 사용자 간 트랜잭션의 유효성을 확인함으로써 안전성, 정확성, 투명성을 확보한다 [6].

2. 관련 연구

2.1 Open Zeppelin

오픈 제플린은 안전한 스마트 컨트랙트를 개발하기 위해 사용되는 라이브러리이다 [7]. 오픈 제플린의 라이브러리 중 'ERC20'을 상속하면 ERC-20 토큰의 이름, 심볼, 허용할 소수점 자릿수 등을 설정하고 발행할 수 있다. 또한,

발행된 토큰의 총량 확인, 입력한 주소에 대한 토큰 잔량 확인, 토큰 전송 등 토큰에 관련된 함수 사용이 가능하다. 발행된 토큰은 오픈 제플린의 'Crowdsale' 라이브러리를 상속한 뒤 블록체인 네트워크에 배포된 스마트 컨트랙트을 통해 판매할 수 있다. 본 논문에서는 스마트 컨트랙트 기반 토큰 교환 시스템에 사용될 임의의 토큰을 발행하고 구매하기 위해 사용한다.

2.2 Metamask

메타마스크는 브라우저에서 이더리움 개인 지갑을 안전하고 편리하게 관리할 수 있는 크롬 확장 프로그램이다. 메타마스크를 이용하면 전체 이더리움 노드를 실행하지 않고도 브라우저에서 분산형 애플리케이션인 이더리움 Dapp (Decentralized Application)을 바로 실행할 수 있을 뿐만 아니라 여러 사이트에서 유저의 ID를 관리하고 블록체인 트랜잭션에 서명할 수 있는 사용자 인터페이스를 제공하는 장점이 있다 [8]. 본 논문에서는 프론트 엔드에서 요청 할 수 있는 '환전', '사용자 계좌 잔액 확인' 등과 같은 스마트 컨트랙트의 함수에 대해 사용자가 정당한 블록체인 네트워크의 참여자이며, 함수를 실행하기 위한 수수료(Gas)를 지불했음을 트랜잭션에 서명하고 '송금 정보', '수령자 정보' 등과 함께 서명된 트랜잭션을 블록체인 네트워크에 전파한다.

2.3 Web3.js

Web3.js 는 HTTP와 IPC 연결을 사용하여 로컬 또는 원격 이더리움 노드와 상호작용 할 수 있는 라이브러리 모음이다 [9]. 즉, Web3.js를 이용하면 프론트 엔드를 실제 토큰을 사용하는 이더리움 메인 네트워크, 테스트를 위해 발행한 토큰을 사용하는 테스트 넷, 로컬 환경에서 동작하는 로컬 네트워크와 연결할 수 있다. 또한, 블록체인 네트워크에 배포된 스마트 컨트랙트 주소와 스마트 컨트랙트를 인코딩하여 생성된 ABI (Application Binary Interface)와 함께 Web3.js 함수를 사용하면 블록체인 네트워크에 배포된 스마트 컨트랙트의 함수를 호출할 수 있다. 메타마스크와 연동도 가능하며 현재 메타마스크 사용자가 보유한 ERC-20 토큰 종류, 잔액 정보, 트랜잭션 발생 및 결과 확인을 할 수 있다.

3. 스마트 컨트랙트를 이용한 토큰 교환 시스템

본 연구에서는 특정 조건을 만족할 시 자동으로 계약을 이행해주는 스마트 컨트랙트를 이용하여 ERC-20 토큰을 교환하는 분산형 토큰 교환 시스템을 제안한다. 실제 거래 중인 이더리움 토큰을 이용한 연구는 비용이 발생하기 때문에 임의로 발행한 세 개의 ERC-20 토큰을 통해 교환 시스템을 연구한다. 임의로 발행한 토큰들은 환율이 존재하지 않기 때문에 데몬을 이용하여 정해진 시간마다 환율을 변동시키고 스마트 컨트랙트에 환율을 적용할 것이다.

또한, 임의로 발행한 토큰은 암호화폐 거래소에 등록되지 않은 '널리 알려지지 않은 토큰'이 될 수 있다. 즉, 사용자는 분산형 토큰 교환 시스템을 이용하면 중앙화된 암호화폐 거래소에 등록되지 않은 토큰이라도 '토큰의 32자리 주소'와 '토큰 간 환율'을 알고 있다면 토큰 간 교환이 가능하다.

3.1 시스템 아키텍쳐

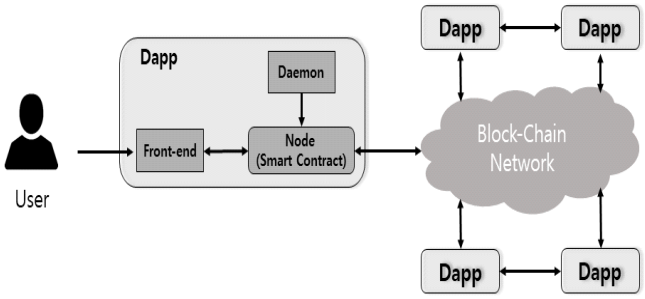
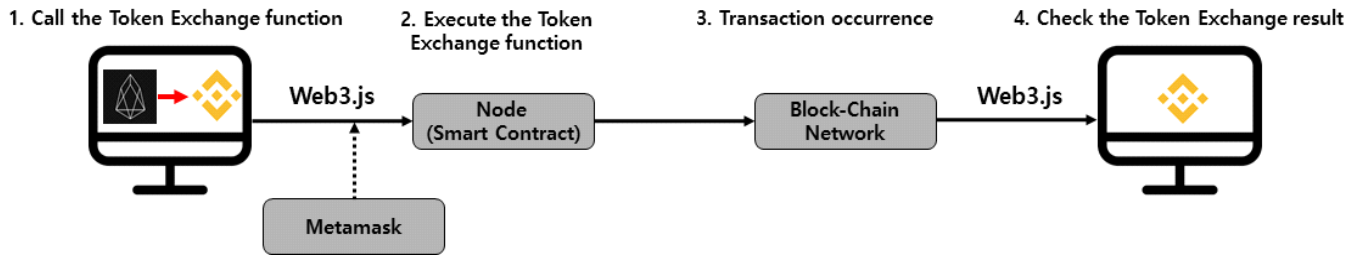


그림 1은 스마트 컨트랙트를 이용한 분산형 토큰 교환 시스템의 시스템 구조를 나타낸다. 암호 화폐의 사용자가 토큰 교환 시스템을 접속하기 위한 '프론트 엔드', 임의의 환율을 변동시키고 이를 스마트 컨트랙트에 설정하는데 필요한 '데몬', 트랜잭션을 저장하고 전파하는 '노드'를 가지고 있으며 토큰 교환과 관련된 함수를 저장하는 '스마트 컨트랙트'가 이더리움 Dapp을 구성한다. 이더리움 Dapp과 블록체인 네트워크가 본 연구에서 제안하는 분산형 토큰 교환 시스템을 구성하는 요소이다. 프론트 엔드에서 요청된 '환전', '사용자 계좌 잔액 확인', 수수료 등과 같은 모든 트랜잭션 정보들은 노드에 저장 후 블록체인 네트워크에 전파된다. 즉, 데이터를 분산하여 저장하고 네트워크 참여자들은 투명하게 기록된 분산형 데이터의 장부를 확인할 수 있다

제안하는 분산형 토큰 교환 시스템은 다음과 같이 동작한다.

임의의 토큰을 발행하고 사용자가 구매할 수 있도록 오픈 제플린 라이브를 이용하여 Crowdsale을 위한 스마트 컨트랙트를 작성한다. 작성된 스마트 컨트랙트는 브라우저 기반 Solidity 컴파일러인 Remix IDE에서 일정량의 수수료인 Gas를 지불하면 블록체인 네트워크에 배포된다. Gas는 소량의 이더(Ether)이며 무분별한 EVM(Ethereum Virtual Machine) 사용을 방지한다. 이렇게 배포된 스마트 컨트랙트는 블록에 저장되기 때문에 수정이 불가하다.

이 스마트 컨트랙트의 32자리 주소에 블록체인 네트워크에 참여한 사용자가 이더리움 네트워크의 화폐 단위인 이더(Ether)를 지불하면 사용자는 Crowdsale을 위한 스마트 컨트랙트 내에 발행된 해당 토큰 양만큼 구매하고 소유할 수 있다. 여기에서, 1 Wei는 10^{-18} ETH 이다. 게다가, 새로이 구매하는 토큰의 양은 1 Wei 당 해당 토큰의 수와 지불한 이더의 양을 곱함으로써 결정된다.



(그림 2) 토큰 환전 절차

Crowdsale을 위한 스마트 컨트랙트와 별도로 토큰 교환을 위한 스마트 컨트랙트도 블록체인 네트워크에 배포되어야 한다. 이 스마트 컨트랙트에는 데몬을 통하여 생성된 환율이 일정 시간마다 갱신되며 저장된 환율 값은 프론트 엔드에서 사용자에게 표시된다.

<표 1> 스마트 컨트랙트를 이용한 토큰 교환 시스템 기능

Target	Name	Description
User	Exchange	토큰 교환
	Exchange and Transfer	토큰 교환 후 송금
	Check Balance	입력 계좌 토큰양 확인
	Check Transaction	입력 계좌 트랜잭션 확인
Admin	Deposit	스마트 컨트랙트에 토큰 입금
	Withdraw	스마트 컨트랙트로부터 토큰 출금
	Check Smart Contract Balance	스마트 컨트랙트 토큰양 확인

사용자는 토큰을 가지고 있다면 스마트 컨트랙트를 이용한 프론트 엔드에서 [표 1]에서 정의하는 'User' 기능들을 사용할 수 있다.

스마트 컨트랙트에서는 토큰 교환을 위한 충분히 많은 수의 토큰들을 보유하고 있어야 한다. 또한, 교환 시 발생한 수수료를 스마트 컨트랙트에서 관리자 계좌 주소로 출금할 수 있어야 한다. 이와 같은 기능을 위하여 'Admin' 기능이 부가적으로 존재한다. 만약 현재 메타마스크의 계좌가 스마트 컨트랙트를 최초에 배포한 32자리 사용자 계좌 주소와 같다면 관리자를 위한 페이지로 이동할 수 있다. 보안을 위해 파이어베이스 인증 시스템을 사용하였으며 정상적으로 관리자 ID, PASSWORD로 로그인을 하였다면 스마트 컨트랙트에 토큰 관리를 위한 'Admin' 기능을 활용할 수 있게 된다.

데몬에서는 교환을 위한 스마트 컨트랙트의 32자리 주소와 ABI, 토큰의 32자리 주소 그리고 Web3.js를 이용하여 스마트 컨트랙트에 임의로 발생한 환율을 설정한다. 스마트 컨트랙트를 작성하기 위한 Solidity 언어는 이더리움 내에서 사용하는 암호 화폐인 이더(Ether)를 다루기 때문에 수를 정확하게 표현하지 못하는 부동 소수점 타입을 지원하지 않는다. 이와 같은 이유로 데몬에서 발생한 임의의 환율에 10^8 을 곱하여 소수점을 정수로 변경한 뒤 스마트 컨트랙트에 저장한다. 스마트 컨트랙트에 저장된 환율을 프론트 엔드에서 사용할 시에는 역으로 10^{-8} 을 곱하여 사용한다.

3.2 토큰 환전 절차

스마트 컨트랙트 기반 분산형 토큰 교환 시스템에 있어서 토큰 환전 절차는 [그림 2]에 도시된 방식처럼 이루어져 있다. 토큰 환전 절차를 수행하는 데 있어서, 스마트 컨트랙트에 토큰 교환 및 사용자의 환전을 위한 충분히 많은 토큰이 존재함을 가정한다. 제시하는 토큰 환전 절차는 다음처럼 4개의 절차로 구성되어 있다.

1. 환전을 위한 함수 호출 : 프론트 엔드에서는 환전을 진행할 사용자 계좌, 환전 전 토큰 주소, 환전 후 토큰 주소, 환전하고 싶은 양을 입력받는다. 스마트 컨트랙트에서 두 토큰 사이의 환율을 가져와 환전하고 싶은 양과 곱하여 환전 받는 양 (추정치)를 자동으로 계산하여 나타낸다.
환전 정보를 입력 후, 환전 진행을 누르면 프론트 엔드에 표시된 환율을 스마트 컨트랙트 함수 인자로 사용한다. 데몬은 일정 시간마다 변동된 환율을 스마트 컨트랙트에 저장하기 때문에 프론트 엔드에서 요청한 환율과 스마트 컨트랙트가 현재 저장하고 있는 환율과 다를 수 있다. 두 환율의 차이가 스마트 컨트랙트에서 설정한 허용 가능한 차이 값인 'Threshold'보다 작다면 환전을 진행한다.
토큰 교환을 위해서는 상속한 ERC20의 approve, transferFrom, transfer 함수를 호출해야 한다. approve 함수는 spender가 원하는 양만큼 토큰을 사용 할 수 있도록 허용하고, transferFrom, transfer 함수를 통해 허용된 양만큼 토큰을 전송할 수 있다. 함수는 Web3.js를 통해 호출하며 트랜잭션의 서명을 위해 메타마스크를 이용한다.

2. 스마트 컨트랙트 환전 함수에서는 사용자 계좌 주소가 유효한지 확인 후 Threshold 차이를 확인한 뒤, 환전을 진행할 사용자 계좌로부터 토큰을 transferFrom 함수를 통해 가져온다. 그리고 환전 후 토큰을 사용자 계좌로 transfer 함수를 통해 보낸다. 그 뒤 transfer 함수의 이벤트를 emit 하여 트랜잭션 상태를 추적한다. 만약 사용자 계좌 또는 교환을 위한 스마트 컨트랙트에 충분한 양의 토큰을 가지고 있지 않으면 트랜잭션은 실패한다.

3. 스마트 컨트랙트로부터 실행된 함수들은 메타마스크에서 함수 호출 시 입력한 Gas의 양과 연결된 블록체인 네트워크의 상태에 따라 트랜잭션의 속도가 달라진다. 프론트 엔드에서는 비동기 방식을 이용하여 스마트 컨트랙트 함수들을 차례대로 진행한다.

4. 트랜잭션 호출이 정상적으로 진행되었다면 Polling 방식을 통해 지속해서 트랜잭션 상태를 확인한다. 트랜잭션이 성공적으로 완료되었다면 최종적으로 환전된 내용을 프론트 엔드에 보여주게 된다. 이 경우에 있어서, [그림 3]과 [그림 4]는 각각 토큰 환전 전과 후의 결과를 보여준다.

정보	값
환전 지갑 주소	0xd17a1BCC63cEc3Bb0459fdA1F43171e6Be66055A
보낸 토큰 종류	EOS
보낸 토큰 양	10.
수수료	0.160579314 BNB

(그림 3) 토큰 교환 전

정보	값
받는 계좌 주소	0xd17a1BCC63cEc3Bb0459fdA1F43171e6Be66055A
받은 토큰 종류	BNB
받은 토큰 양	53.365858685999996

(그림 4) 토큰 교환 후

4. 결론 및 향후 연구

본 논문에서는 이더리움 플랫폼에 배포된 스마트 컨트랙트 (Smart Contract)를 이용한 분산형 토큰 교환 시스템 시스템을 제안하였다. 블록체인에 배포된 스마트 컨트랙트를 이용함으로써 안전성을 확보하였으며 사용자가 직접 교환하려는 토큰의 주소를 입력하기 때문에 널리 알려지지 않은 토큰에 대해서도 토큰 교환이 가능하다. 또한, 토큰 교환을 목적으로 하는 시스템이기 때문에 토큰 교환을 위한 명확한 인터페이스를 가지며 '사용자 보유한 토큰 확인' 등과 같은 부가 기능들을 사용할 수 있다. 게다가, 중앙화된 암호 화폐 거래소보다 인건비, 운영비 등을 줄일 수 있어 더 낮은 수수료를 지불하고 토큰 교환 시스템을 이용할 수 있을 것으로 기대한다.

향후 연구 계획은 이더리움의 ERC-20 토큰 간 교환에서 나아가 스마트 컨트랙트를 이용하여 이더리움 외 블록체인 플랫폼의 코인 및 토큰과 교환할 수 있는 크로스 체인 시스템을 연구하는 것이다.

이와 더불어 스마트 컨트랙트 기반 분산형 토큰 교환 시스템을 Chrome Extension으로 사용할 수 있도록 연구하여 사용자의 편의성을 극대화하는 것이 목표이다.

5. 감사의 글

"본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW중심대학지원 사업의 연구 결과로 수행되었다 (2017-0-00096)."

참고문헌

[1] N. Satoshi, "Bitcoin: A Peer-to-Peer Electronic Cash System." 2008.
[2] V. Buterin, "A next-generation smart contract and decentralized application platform", Ethereum White paper, 2014
[3] 암호화폐 시가총액, https://coinmarketcap.com/ko/
[4] etherscan.io, https://etherscan.io/tokens
[5] 김경애, "빗썸 해킹 원인, 비정상적 출금 시도? 공격 경로 '주목'", 보안뉴스, 2018년06월 20일 자
[6] 이종훈, 윤성욱, "블록체인에서 스마트컨트랙트 관련 연구", *한국정보과학회 학술발표논문집* pp. 2,091-2,093, 2019년
[7] 오픈 제플린, https://openzeppelin.com
[8] 메타마스크, https://metamask.io
[9] Web3.js, https://web3js.readthedocs.io