JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 25, No. 4, pp. 1087-1094, Apr. 2024

Check for updates

스마트 컨트랙트 보안 취약점 분석 및 보안 자동화 모델

조 도 은*
*목원대학교 SW교양학부 교수

Smart Contract Security Vulnerability Analysis and Security Automation Model

Do-Eun Cho*
*Professor, Department of SW Liberal Arts, Mokwon University, Daejeon 35349, Korea

[요 약]

스마트 컨트랙트는 블록체인 기술의 핵심 구성요소로 계약 조건이 충족되면 자동으로 시행되기 때문에 디지털 데이터에 대한 신뢰도가 필요한 다양한 분야에서 활용되고 있다. 그러나 스마트 컨트랙트는 여러 보안 취약점과 프라이버시 위협을 가지고 있으므로 이에 대한 보안 강화 방안이 필요하다. 본 연구에서는 스마트 컨트랙트의 동작 과정별 보안 취약점을 분석하고, 각 단계별 보안 자동화 모델을 제시하였다. 스마트 컨트랙트의 보안은 전체 생명 주기에 걸쳐 중요하며, 각 동작 단계에서 수행되어야 할 보안 사항을 자동화한다면 오류를 최소하며, 보안 유지 관리의 부담을 줄일 수 있다. 제안한 보안 자동화 모델의 보안성 분석을 통해 각 단계별 보안 위협에 대한 보안성을 모두 만족하는 것으로 분석하였다. 향후 제안한 모델을 다양한 실제 사례에 적용하여 구현함으로써 블록체인 기술의 보안 및 신뢰성을 강화하고, 광범위한 사용자에게 안전한 블록체인 환경을 제공할 수 있을 것이다.

[Abstract]

Smart contracts are a key component of blockchain technology and are used in various fields that require reliability for digital data because they are automatically implemented when contract conditions are met. However, smart contracts have several security vulnerabilities and privacy threats; thus, measures to strengthen security are needed. In this paper, security vulnerabilities for each operation process of smart contracts were analyzed, and a security automation model for each step was presented. The security of smart contracts is important throughout its entire life cycle, and automating the security steps to be performed at each operation stage can minimize errors and reduce security maintenance. A security analysis of the proposed security automation model revealed that all security requirements against threats at each stage were satisfied. By applying and implementing the proposed model in the future to various real-world cases, it will be possible to strengthen the security and reliability of blockchain technology and provide a safe blockchain environment to a wide range of users.

색인어 : 블록체인 보안, 스마트 컨트랙트, 보안 취약점, 프라이버시, 보안 모델
Keyword : Blockchain Security, Smart Contract, Security Vulnerabilities, Privacy, Security Model

http://dx.doi.org/10.9728/dcs.2024.25.4.1087

Received 03 March 2024; **Revised** 14 March 2024
Accepted 21 March 2024

***Corresponding Author; Do-Eun Cho**
Tel: +82-42-829-7649
E-mail: decho@mokwon.ar.kr

Ⅰ. 서 론

블록체인 기술은 디지털 거래의 투명성과 무결성을 제공하는 혁신적인 기술이다. 스마트 컨트랙트는 블록체인 기술의 핵심 구성 요소로, 계약 조건이 충족될 때 자동으로 계약 내용을 이행하도록 설계된 자동 실행 프로그램 또는 알고리즘이다. 스마트 컨트랙트와 블록체인은 금융, 부동산, 의료, 공급망 관리, 투표 시스템 등 다양한 분야에서 중요한 역할을 한다. 스마트 컨트랙트에는 트랜잭션에 대한 모든 정보가 포함되어 있으며 기존 계약과는 다르게 제 3자가 관여하지 않고 암호화된 거래를 참여자 간 공유하기 때문에 신뢰성과 투명성이 보장된다. 그러나 스마트 컨트랙트는 여러 보안 취약점과 프라이버시 위협에 직면해 있다. 스마트 컨트랙트는 한 번 블록체인에 배포되면 수정할 수 없어 이를 악용한 악의적인 공격이 발생하고 있다. 특히, 공개적으로 접근 가능한 블록체인 네트워크에서 이러한 문제는 중대한 위험을 야기할 수 있다. 2016년에 발생한 '더 다오(DAO)' 해킹 사건은 스마트 컨트랙트의 취약점을 악용할 경우 심각한 금융적 손실을 야기할 수 있음을 보여주는 대표적인 사례이다[1]. 이 사건에서 약 5천만 달러 상당의 이더리움이 도난당했다. 더구나 스마트 컨트랙트 내의 데이터가 블록체인에 영구적으로 기록되므로 개인 정보가 포함된 경우 이를 적절히 보호하지 못할 위험이 있다.

본 연구의 목적은 블록체인 기반 스마트 컨트랙트의 안전한 활용을 위하여 보안 취약점을 분석하고, 이를 위한 보안 자동화 모델을 연구하는 것이다. 스마트 컨트랙트의 보안 취약점은 시스템 전체의 신뢰성에 영향을 미칠 수 있으며, 개인정보의 노출과 같은 프라이버시 문제는 법적, 윤리적 문제를 야기할 수 있다. 따라서 본 연구는 현재의 스마트 컨트랙트의 동작 과정 별 보안 위협을 분석하고, 이를 해결할 수 있는 방안을 모색함으로써 블록체인 기술의 안전한 활용에 기여하고자 한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서 이더리움 블록체인의 구조와 스마트 컨트랙트의 동작과정을 소개하고, 3장에서 스마트 컨트랙트의 동작 과정별 보안 취약점에 대해 상세히 설명한다. 4장에서 스마트 컨트랙트의 보안을 강화하기 위한 보안 자동화 모델을 제안하고, 보안성을 분석한다. 마지막으로 5장에서 결론 및 향후 연구과제를 제시한다.

Ⅱ. 관련 연구

2-1 이더리움 블록체인과 스마트 컨트랙트

블록체인은 여러 컴퓨터에 분산되어 데이터를 저장하는 기술로, 각 데이터 블록이 이전 블록에 대한 암호화된 참조를 포함하여 '체인'을 형성한다. 이 구조는 데이터의 무결성과 투명성을 보장하며, 중앙집중식 서버 없이도 정보의 안전성을 유지한다[2].

스마트 컨트랙트는 블록체인의 불변성, 분산성, 투명성을 활용하여 신뢰할 수 있고 변경 불가능한 방식으로 계약 조건을 자동으로 실행한다[3]. 이더리움과 같은 최신 블록체인 플랫폼은 튜링 완전성(Turing-completeness)을 갖춘 스마트 컨트랙트를 통해 금융 서비스, 의료, 부동산 거래, 교육, 디지털 ID, 투표 시스템 등 다양한 분야에서 혁신적인 응용 프로그램 개발을 가능하게 한다[4].

이더리움 블록체인은 표 1과 같이 여러 계층으로 구성되어 있다. 사용자가 직접 상호 작용하는 응용 계층(Application Layer), 네트워크의 모든 참가자의 합의 메커니즘이 구현된 합의 계층(Consensus Layer), 블록 체인의 데이터 구조를 관리하는 데이터 계층(Data Layer), P2P(Peer-to-Peer) 네트워크 연결을 통해 데이터를 전파하는 네트워크 계층(Network Layer)이다[5]. 이더리움 블록체인 블록 구조를 살펴보면 그림 1과 같다. 각 블록은 이전 블록의 해시, 타임스탬프, 머클 루트(Merkle Root), 논스(Nonce)로 구성된다. 각 블록은 이전 블록의 해시를 사용하여 다른 블록과 연결된다. 타임스탬프는 각 블록이 생성되는 시간으로 트랜잭션의 순서를 추적하고 검증하는데 사용된다.

표 1. 이더리움 블록체인의 계층적 구조
Table 1. Layered structure of Ethereum blockchain

Layer	Description	Components
Application Layer	Where users interact with smart contracts and DApps.	Smart Contracts, DApps, Business Logic, Chain Code
Consensus Layer	Ensures all nodes agree on the state of the blockchain.	Proof of Work, Proof of Stake, Consensus Algorithms
Data Layer	Manages the blockchain's data and ensures its integrity and security.	Digital Signatures, Hash, Merkle Trees, Transactions
Network Layer	Facilitates data propagation and communication across the blockchain network.	P2P Connections, Network Protocols

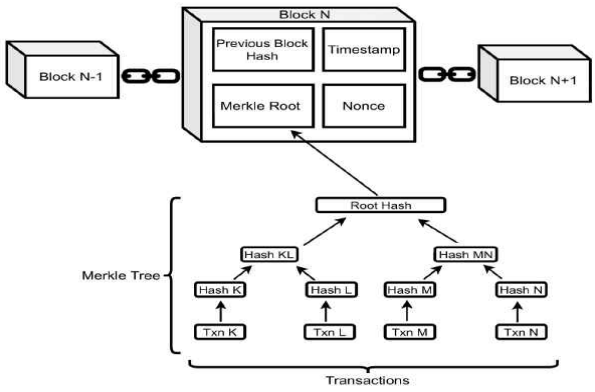

그림 1. 이더리움 블록체인 블록 구조[5]
Fig. 1. Struct of an Ethereum blockchain block

머클 트리는 모든 트랜잭션의 해시를 요약하여 하나의 해시값으로 집약한다. 논스는 작업 증명(Proof of Work)과 합의 알고리즘에서 사용하는 값이다.

2-2 스마트 컨트랙트의 동작 과정

스마트 컨트랙트는 블록체인 안에 컴퓨터 코드로 프로그램된 분산된 합의로 정의되며, 계약 이행 요청 시 미리 프로그램된 계약 조건에 따라 계약 내용을 수행한다[6]. 대부분의 블록체인 시스템은 스마트 컨트랙트의 등록, 삭제 기능을 제공하고 있으며, 블록체인에 등록된 스마트 컨트랙트를 트랜잭션을 통해 호출함으로써 계약 이행을 요청할 수 있다. 그림 2는 스마트 컨트랙트의 동작 과정을 설계 및 배포, 트랜잭션 전송, 실행과 기록으로 3단계로 구분하여 나타낸 것이다.

1) 스마트 컨트랙트 설계 및 배포

스마트 컨트랙트의 기본 구조와 기능이 설계되는 단계로, 개발자들은 스마트 컨트랙트가 수행할 작업, 처리할 데이터의 종류, 스마트 컨트랙트의 실행 조건을 정의한다. 작성된 스마트 컨트랙트는 마이너들(Miners)에 의해 블록체인 네트워크에 배포된다. 이더리움에서는 솔리디티(Solidity) 언어를 사용하여 스마트 컨트랙트를 프로그래밍한다. 프로그래밍 된 스마트 컨트랙트는 솔리디티 컴파일러(solc)를 통해 바이트코드로 변환되며, 이 바이트코드는 블록에 포함되어 이더리움 가상 머신(EVM)에서 실행된다. EVM은 이더리움 스마트 컨트랙트의 바이트코드를 실행하는 32바이트 스택 기반의 실행 환경이다[7],[8].

2) 사용자 트랜잭션 전송

사용자가 스마트 컨트랙트와 상호작용하기 위해서는 트랜잭션을 전송한다. 이 트랜잭션은 스마트 컨트랙트의 특정 함수를 호출하거나, 특정 조건을 충족시키기 위한 데이터를 포함할 수 있다. 사용자는 블록체인 네트워크에 트랜잭션을 전송하고, 이는 블록체인 노드들에 의해 검증되고 블록에 포함된다. 이때 조건이 충족되면, 스마트 컨트랙트는 자동으로 계약 조항을 이행한다. 이 과정은 완전히 자동화되어 있어 사람의 개입이 필요 없다.

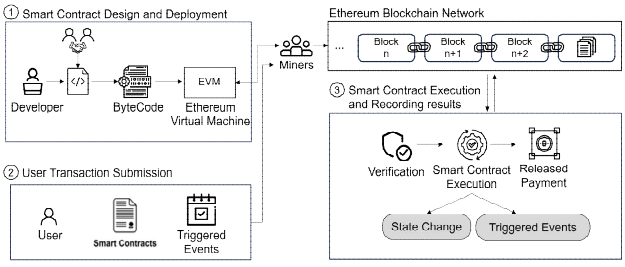

그림 2. 스마트 컨트랙트 동작 과정
Fig. 2. Smart contract operation process

3) 스마트 컨트랙트 실행과 기록

블록체인 네트워크에 전송된 트랜잭션은 스마트 컨트랙트에 의해 자동으로 처리된다. 'State change'는 함수 실행의 결과로 스마트 컨트랙트 계약의 데이터나 상태가 변경되는 것을 나타낸다. 'Triggered Events'는 상태 변경에 따라 스마트 컨트랙트에서 이벤트가 트리거 되는 것을 나타낸다. 이벤트는 계약 외부의 참여자들에게 특정 사건이 발생했음을 알릴 때 사용된다. 스마트 컨트랙트에 의한 모든 트랜잭션 결과는 블록체인에 기록된다. 이는 블록체인의 무결성과 탈 중앙화된 특성을 통해 보장된다. 트랜잭션이 완료되면 트랜잭션과 관련된 가스 비용이 계산 및 지불되고, 네트워크를 통해 비용 지불 결과를 표시한다. 따라서 이 단계에서 블록체인 네트워크는 트랜잭션의 결과를 검증하고, 블록에 포함시켜 네트워크 전체에 분산시킨다.

이러한 동작 과정은 스마트 컨트랙트가 효율적으로 운영되고 관리되기 위해 필수적이며, 각 단계는 블록체인의 보안 및 투명성을 유지하는 데 중요한 역할을 한다.

Ⅲ. 스마트 컨트랙트의 보안 취약점 분석

스마트 컨트랙트는 자동화된 계약 실행을 통해 블록체인 기술의 가장 혁신적인 응용 중 하나로 자리 잡았지만, 보안 및 프라이버시에 관한 취약점으로 인해 여러 위험에 노출되어 있다. 이러한 취약점은 스마트 컨트랙트의 신뢰성과 사용자의 프라이버시를 위협한다.

3-1 스마트 컨트랙트 설계 및 배포 단계의 취약점

스마트 컨트랙트 설계 및 배포에서 보안위협 사항으로는 코드 취약점, 프라이버시 노출 등이 있다.

- 코드 취약점: 프로그래밍 오류나 논리적 결함으로 인해 스마트 컨트랙트에 보안 취약점이 생길 수 있고, 이러한 취약점을 통해 공격자가 시스템을 악용할 가능성이 있다. 예를 들어, '재진입 공격(Reentrancy Attack)'은 스마트 컨트랙트내의 동일한 함수를 여러 번 호출하여 예상치 못한 방식으로 동작하게 만들 수 있는 취약점이다. 또한 ' 오버플로 공격(Overflow Attack) '은 변수가 허용하는 최대값을 초과하여 데이터를 손상시킬 수 있다. 이를 통해 시스템을 조작하고 부당한 이득을 취할 수 있다.
- 프라이버시 노출: 스마트 컨트랙트는 모든 거래 내용을 블록체인에 공개적으로 기록한다. 이는 투명성을 제공하지만, 동시에 사용자의 민감한 정보가 외부에 노출될 수 있는 위험을 안고 있다. 이는 사용자 데이터의 노출 및 오용 가능성으로, 스마트 컨트랙트가 민감한 정보를 처리할 때 발생할 수 있다. 더구나 스마트 컨트랙트가 실행될 때 블록체인에 개인정보가 기록될 수 있으며, 블록체

인의 불변성 특성으로 인해 한 번 기록된 데이터는 삭제되지 않는다. 이로 인해 개인정보 보호와 데이터 파기에 관한 문제가 발생할 수 있다.

스마트 컨트랙트의 오류를 검출하기 위한 기존 연구들은 다양한 방식으로 이루어져 왔다[9]-[11]. 이 연구들은 스마트 컨트랙트 설계 단계에서 발생할 수 있는 오류들을 식별하며, 이러한 오류들을 방지하기 위한 다양한 방법들을 제안하고 있다. 오류 검출기법은 정적 분석, 동적 분석, 형식적 검증, 퍼지 테스팅, 그리고 코드리뷰 및 감사 방법으로 구분된다. 정적 분석은 코드의 구문과 구조를 검사하는 방법으로 Slither, Mythril, Securify 등이 있다. 동적 분석은 테스트 환경에서 스마트 컨트랙트 코드를 실행하여 오류나 취약점을 찾는 기법으로 정적 분석에서 놓칠 수 있는 실행시간 오류를 탐지할 수 있다. 이러한 기법에는 Ganache, Truffle, Remix 등이 있으며, 이들은 개발 환경에서 스마트 컨트랙트를 배포하고 테스트하여 문제를 식별할 수 있다. 형식적 검증은 수학적 방법을 사용하여 스마트 컨트랙트 코드가 특정 명세나 속성을 만족하는지 증명하는 기법으로 복잡한 로직이나 상태를 정확하게 분석할 수 있다. 그 외에 Echidna, Harvey 등 자동화된 퍼지 테스트 기법이 있으며, 코드리뷰 및 감사 방법은 전문가가 코드를 직접 검토하여 논리적 오류나 설계 결함을 찾는 방법이다[12]-[14]. 프라이버시를 보호를 위해서는 제로 지식 증명(Zero-Knowledge Proofs), 동형 암호화, 다자간 계산(MPC) 방안[15]이 활용될 수 있다. 더불어 프라이버시 보호를 위한 설계 원칙을 적용하는 기법으로 사이드체인 및 오프체인 솔루션[16], 프라이버시를 보장하는 암호화 기술을 내장한 스마트 컨트랙트 개발 등의 방안이 사용될 수 있다.

3-2 사용자의 트랜잭션 전송단계의 취약점

사용자가 스마트 컨트랙트와 상호작용하는 단계로, 사용자는 스마트 컨트랙트의 특정 기능을 실행하기 위해 트랜잭션을 블록체인 네트워크로 전송한다. 이때 전송된 트랜잭션은 인증 노드들로 전송된다. 트랜잭션을 전송받은 인증 노드들이 적합성을 인증하게 되면 해당 스마트 컨트랙트를 호출하게 된다. 이 단계에서는 피싱 공격, 네트워크 스니핑, 트랜잭션 조작 등의 보안 위협이 있을 수 있다.

- 피싱 공격: 사용자를 속여 중요 정보(예: 비밀번호, 개인키)를 획득하려는 사기성 공격이다. 피싱 공격에 의해 사용자의 민감한 정보가 노출되는 위험이 있고, 이러한 정보 노출은 스마트 컨트랙트 계정이 해킹당할 가능성을 높일 수 있다.
- 네트워크 스니핑: 사용자가 트랜잭션을 전송할 때, 공격자가 데이터를 가로채는 네트워크 기반의 공격이다. 이를 통해 트랜잭션 내용, 개인 정보 등을 빼낼 수 있다. 이때 데이터가 암호화되지 않은 채 전송되면, 중요한 정보가 쉽게 노출될 수 있다.
- 트랜잭션 조작: 공격자가 트랜잭션 내용을 변경하여 스마트 컨트랙트의 동작을 의도와 다르게 조작하는 행위이다. 이는 금융적 손실이나 스마트 컨트랙트의 잘못된 실행을 초래할 수 있다.

트랜잭션 전송단계에서의 보안 기법은 트랜잭션의 무결성, 기밀성, 그리고 인증성을 보장하는 데 중점을 둔다. 이에 대한 보안 기법은 이미 여러 분야에서 연구 및 활용되고 있다. SSL/TLS와 같은 안전한 프로토콜을 사용하여 데이터를 암호화하고 데이터의 기밀성과 무결성을 보장해야 하며, 트랜잭션 발신자에 의해 디지털 서명이 필요하다. 발신자는 자신의 개인키를 사용하여 트랜잭션에 서명하고, 이 서명은 트랜잭션의 무결성과 발신자의 인증을 보장하는 데 사용된다. 사용자 인증의 강도를 높이기 위해 멀티팩터 인증(MFA; Multi-Factor Authentication) 기법을 사용할 수 있다.

3-3 스마트 컨트랙트 실행 및 기록 단계의 취약점

블록체인 네트워크에 전송된 트랜잭션은 스마트 컨트랙트에 의해 자동으로 처리된다. 이때 스마트 컨트랙트의 코드에 따라 정해진 로직이 실행되며, 수행 결과가 유효하면 해당 결과가 블록체인에 기록된다. 이 단계에서 스마트 컨트랙트 실행 시 가스 한도 문제, 타임스탬프 조작 및 무단 접근의 위협이 발생할 수 있다. 네트워크에 전송된 트랜잭션이 실행 완료되면, 트랜잭션 실행 결과는 합의 프로토콜에 따라 네트워크 참여자(또는 네트워크 노드)에게 전송되며, 유효한 경우 해당 트랜잭션이 블록체인 원장에 추가된다. 이 단계에서는 블록체인 데이터 변조, 불법적 접근 및 해킹, 데이터 무결성 위협이 발생할 수 있다.

- 가스 한도 문제: 블록체인에서 스마트 컨트랙트 실행에는 '가스(Gas)'라고 불리는 단위가 소모된다. 가스 한도 문제는 스마트 컨트랙트 실행에 필요한 가스가 한도를 초과하여 작업이 완료되지 못하는 문제이다. 이는 스마트 컨트랙트 실행이 중단되며, 서비스 거부(DoS) 공격의 원인이 될 수 있다.
- 타임스탬프 조작: 스마트 컨트랙트가 블록의 타임스탬프에 의존할 때, 타임스탬프 조작으로 스마트 컨트랙트의 실행 결과에 영향을 미치는 행위이다. 이를 이용하여 특정 트랜잭션이 유리하게 처리되거나 특정 조건이 만족되는 등의 문제가 발생할 수 있다.
- 무단 접근 및 권한 남용: 스마트 컨트랙트의 기능에 접근 권한이 없는 사용자가 스마트 컨트랙트를 실행하거나, 권한이 있는 사용자가 그 권한을 남용하는 경우이다. 이러한 무단 접근이나 권한 남용으로 인해 민감한 데이터가 노출되거나 잘못된 트랜잭션이 실행될 수 있다.
- 블록체인 데이터 변조: 블록체인에 기록된 데이터가 불법적으로 변경되거나 삭제되는 행위이다. 블록체인의 무결성이 손상되면, 저장된 정보의 신뢰성이 저하된다. 데이터 변조는 트랜잭션 기록의 진실성을 손상시키고, 블록체인 네트워크 전체의 신뢰도를 떨어뜨릴 수 있다.

스마트 컨트랙스 실행 및 결과 기록 단계의 보안 기법은 다음과 같다. 우선, 스마트 컨트랙트 코드의 충분한 테스트 및 시뮬레이션을 통해 오류를 미리 발견하도록 하고 수정해야 한다. 이는 코드의 버그를 최소화하고, 가스 소모를 최적화하여 네트워크 자원의 효율적 사용을 보장한다. 또한 스마트 컨트랙트의 실행 효율성을 높이기 위해 코드 최적화 작업이 필요하다. 타임 스탬프 조작 방지를 위해서는 블록체인의 합의 메커니즘과 규칙을 활용한다. 블록체인 네트워크는 타임 스탬프 조작을 어렵게 만드는 분산된 합의 프로세스를 가지고 있으며, 이는 네트워크의 무결성을 유지하는 데 중요한 역할을 한다. 스마트 컨트랙트의 중요한 기능에 대한 접근을 제한하기 위해서는 역할 기반의 액세스 제어(RBAC)기법을 적용하거나 접근 제어목록(ACL)을 구현한다. 이를 통해 무단 접근 및 권한 남용을 방지하고, 스마트 컨트랙트의 민감한 기능을 보호한다. 또한 스마트 컨트랙트 실행 결과와 중요한 상태 변경 사항은 이벤트 로그를 통해 기록된다, 이 로그 정보의 무결성과 기밀성을 보장하기 위해 로그 데이터의 안전한 처리와 저장이 필요하다.

3-4 STRIDE를 적용한 스마트 컨트랙트 취약점

위에서 언급한 스마트 컨트랙트의 보안 취약점을 STRIDE 위협 모델링 기법[17]을 적용하여 보안 위협 사항을 분석하였다. 표 2는 STRIDE 분류에 의한 스마트 컨트랙 취약점을 나타낸 것이다.

Ⅳ. 스마트 컨트랙트의 보안 자동화 모델

4-1 스마트 컨트랙트 설계 및 배포 단계의 보안 모델

스마트 컨트랙트의 설계 및 배포 과정에서 고려해야 할 보안 사항을 자동화하여 개발자가 실수로 보안 취약점을 발생하는 것을 방지하고, 스마트 컨트랙트가 배포되기 전에 잠재적인 문제를 식별하여 해결할 수 있게 한다. 그림 3은 스마트 컨트랙트 설계 및 배포 단계에서 보안 자동화 모델의 동작과정을 나타낸 것이다.

개발자는 스마트 컨트랙트에서 처리될 민감한 데이터를 식별하고, 이에 대한 데이터 처리 방법 및 프라이버시 정책을 결정한다, 이 데이터는 데이터베이스(DB)에서 암호화하여 저장된다. 또한 개발자는 누가 어떤 데이터에 접근할 수 있는지 접근 제어 목록(ACL)을 정의하고, 이를 인증서버(Authentication Server)에 적용하여 데이터 접근을 관리한다. 여기서 역할기반 액세스 제어(RBAC) 기법을 사용하여 사용자의 역할에 따라 접근 권한을 적용한다. 개발된 스마트 컨트랙트는 감사 도구(Audit Tool)를 사용하여 코드 오류 및 프라이버시 취약점을 검사하고 보안 감사를 수행한다. 테스트 시스템(Test System)을 통해 스마트 컨트랙트가 코드 오류가 없으며 프라이버시 정책 및 관련 법규를 준수하는지 확인한다. 테스트를 마친 스마트 컨트랙트는 블록체인 네트워크에 배포된다. 이 과정에서 민감한 데이터는 블록체인 외부의 데이터베이스에 저장되며, 블록체인 상에서는 참조만 가능하게 한다.

4-2 트랜잭션 전송 단계의 보안 모델

스마트 컨트랙트의 전송단계에서 보안 자동화는 트랜잭션 전송과정을 보다 안전하게 만들며, 다양한 보안 위협으로부터

표 2. STRIDE를 적용한 스마트 컨트랙트 취약점
Table 2. Smart contract vulnerability analysis with STRIDE

STRIDE Element	Smart Contract Vulnerability
Spoofing(S)	– Phishing Attacks: Deceptive attacks aiming to obtain critical information such as passwords and private keys
Tampering(T)	– Code Vulnerabilities: Reentrancy attacks, overflow attacks – Transaction Manipulation: Altering transaction contents to disrupt smart contract operation – Timestamp Manipulation: Affecting the outcome of smart contract execution by manipulating block timestamps – Blockchain Data Tampering: Illegal alteration or deletion of blockchain records
Repudiation(R)	– Reentrancy Attacks: Potential for denial involving unpaid virtual currency, repeated unauthorized transaction requests
Information Disclosure(I)	– Privacy Exposure: Permanent exposure and potential misuse of personal information recorded on the blockchain – Network Sniffing: Exposing transaction contents and personal information – Unauthorized Access and Privilege Abuse: Exposure of sensitive data and execution of incorrect transactions
Denial of Service(D)	– Gas Limit Issues: Smart contract execution halts due to exceeding the gas limit, potentially leading to DoS attacks
Elevation of Privilege(E)	– Unauthorized Access and Abuse of Privileges: Execution of smart contracts by users without access rights or misuse of permissions by authorized users

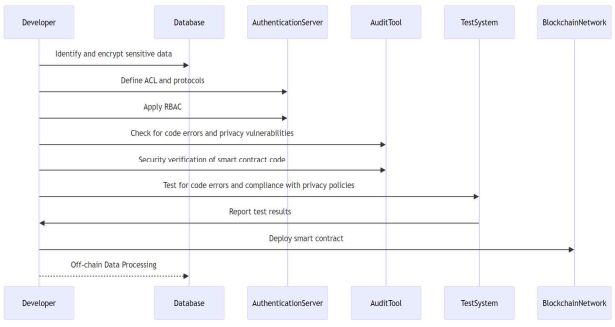

그림 3. 스마트 컨트랙트 설계 및 배포 보안 모델
Fig. 3. Smart contract design and deployment security mode

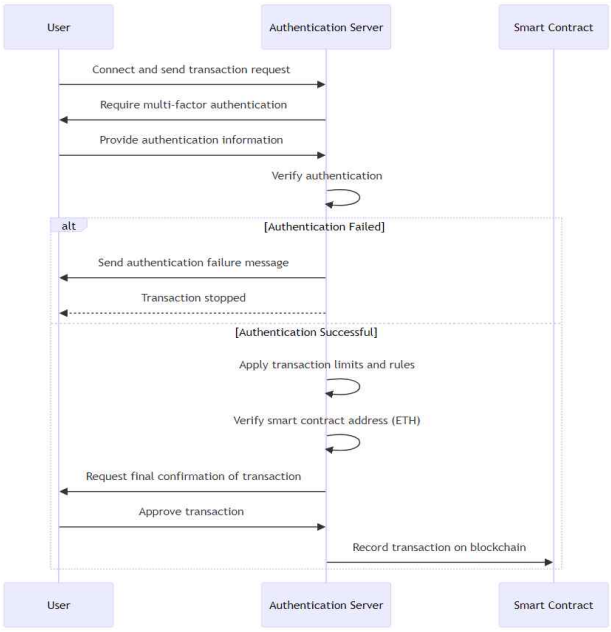

그림 4. 트랜잭션 전송 보안 모델
Fig. 4. Transaction transfer security model

사용자를 효과적으로 보호한다. 이러한 자동화된 보안 접근 방식은 사용자의 실수를 최소화하고, 전체 시스템의 보안 수준을 강화할 수 있다. 그림 4는 트랜잭션 전송 단계에서 보안 자동화 모델의 동작 과정을 나타낸 것이다.

사용자가 트랜잭션을 전송할 때, 전송되는 데이터는 SSL/TLS와 같은 암호화 프로토콜을 통해 보호되어 데이터의 안정성이 확보된다. 사용자가 트랜잭션을 전송하게 되면, 인증 서버는 여러 단계의 인증(예: 비밀번호, OTP, 생체 인증)을 요구한다. 사용자는 이러한 인증 요구 사항에 따라 필요한 정보를 제공하고, 인증서버는 제공된 정보를 검증한다. 인증에 실패할 경우, 사용자에게 인증실패 메시지가 전송되며 해당 트랜잭션은 중단된다. 인증 서버는 또한 사용자 계정의 트랜잭션 한도와 규칙을 적용하여 검토하고, 사용자가 입력한 스마트 컨트랙트 주소(ETH)의 유효성을 확인한다. 모든 검증이 완료되면 해당 트랜잭션은 블록체인에 기록되어 처리된다. 이러한 단계별 과정은 스마트 컨트랙트 트랜잭션의 보안성을 강화하고, 사용자의 실수나 피싱 공격으로부터 중요한 정보를 보호하는 데 핵심적인 역할을 한다.

4-3 스마트 컨트랙트 실행 및 기록 단계의 보안 모델

스마트 컨트랙트 실행 및 기록 단계에서의 보안 자동화는 트랜잭션을 안전하게 수행하기 위해 스마트 컨트랙트를 통해 전송된 트랜잭션을 검증하고, 유효한 트랜잭션만이 최종적으로 수행되도록 하는 것이다. 이 단계에서의 보안 자동화는 스마트 컨트랙트 실행 중 발생할 수 있는 보안 위협을 예방, 감지, 대응하게 한다. 처리가 완료된 트랜잭션은 블록체인에 기록된다. 그림 5는 스마트 컨트랙트 실행 단계에서 보안 자동화 모델의 동작 과정을 나타낸 것이다.

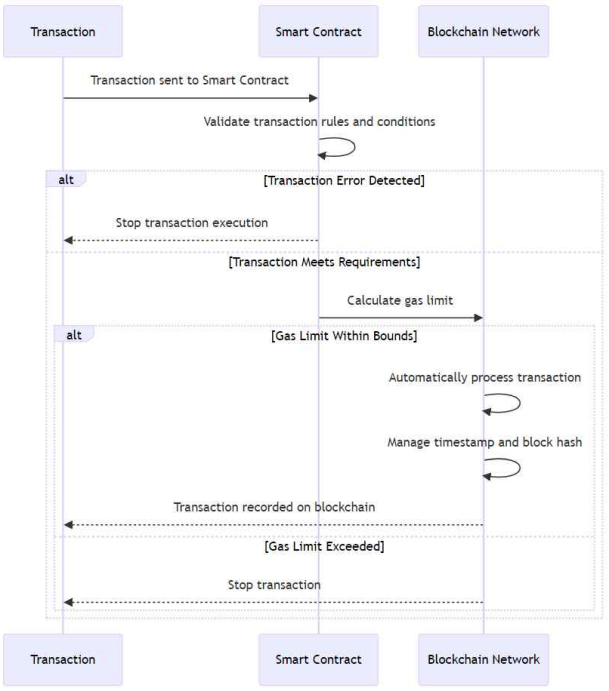

그림 5. 스마트 컨트랙트 실행 및 기록 보안 모델
Fig. 5. Smart contract run and record security model

트랜잭션이 스마트 컨트랙트에 전송되면, 스마트 컨트랙트는 트랜잭션이 설정한 규칙과 조건에 부합하는 지 검증한다. 이 과정에서 오류가 검출되면 트랜잭션 실행은 중단된다. 트랜잭션이 스마트 컨트랙트의 요구사항을 충족하면, 블록체인 네트워크는 트랜잭션의 가스 한도를 계산한다. 가스 한도 내에서 실행 가능한 경우, 트랜잭션은 블록체인 네트워크에 의해 자동으로 처리되며, 타임스탬프와 블록 해시는 합의 메커니즘에 의해 관리된다. 블록체인에 기록된 트랜잭션은 변경할 수 없으며, 모든 블록체인 네트워크 참가자에게 정보의 정확성을 보증한다. 이러한 과정은 스마트 컨트랙트의 실행 중 발생할 수 있는 보안 위협을 예방하고, 트랜잭션의 무결성과 투명성을 보장한다.

4-4 스마트 컨트랙트 보안 모델의 보안성 분석

제안한 스마트 컨트랙트 보안 자동화 모델의 보안성을 스마트 컨트랙트 동작 단계별로 분석하면 다음과 같다.

스마트 컨트랙트의 설계 및 배포단계에서 제안된 보안 모델은 설계 초기 단계에서부터 민감한 데이터를 자동 식별하고 적절한 프라이버시 정책 결정을 통해 개인정보보호를 강화하였다. 또한 접근제어목록(ACL)을 통해 데이터 접근 권한을 세밀히 관리하고, 오프체인(Off-chain) 기법을 적용함으로써 민감한 데이터를 블록체인 네트워크 외부에서 안전하게 처리함으로써 사용자의 개인 정보 보호를 강화하였다. 또한

역할기반 접근제어(RBAC)를 통한 인증 서버는 데이터 접근 권한을 명확히 정의하고 무단 접근을 방지한다. 감사 도구를 사용하여 프라이버시 취약점을 자동으로 검사하며, 테스트 시스템을 통해 프라이버시 준수 여부를 자동으로 테스트 하였다. 따라서 스마트 컨트랙트의 코드 취약점, 프라이버시 노출 등의 보안 위협에 대한 보안성을 모두 만족한다.

사용자의 트랜잭션 전송 단계에서 적용된 보안 모델은 트랜잭션이 전송될 때 SSL/TLS 프로토콜을 사용하여 통신 과정을 암호화함으로써 데이터의 안전성을 보장한다. 또한 인증 서버를 통해 멀티팩터 인증을 수행하여 인증 과정의 강도를 높이고 피싱 공격을 방지한다. 전송된 트랜잭션은 스마트 컨트랙트에 의해 프로그래밍된 로직에 따라 데이터를 검증하며, 비정상 트랜잭션이 감지되면 사용자에게 중단 메시지를 전송하고, 트랜잭션을 정지한다. 이러한 보안 기법은 피싱 공격, 네트워크 스니핑, 트랜잭션 조작 등의 보안 위협에 대응하여 전체적인 보안성을 강화한다.

트랜잭션 실행 및 기록 단계에서 적용된 보안 모델은 스마트 컨트랙트를 통해 트랜잭션이 설정된 규칙과 조건을 충족하는지 검증함으로써 코드에서 발생할 수 있는 잠재적인 문제를 사전에 차단하게 된다. 트랜잭션 처리 전에 블록체인 네트워크에서 가스 한도를 계산하여 트랜잭션이 네트워크에 과도한 부담을 주지 않도록 관리한다. 또한 블록체인 네트워크는 트랜잭션의 타임 스탬프와 블록의 해시 값을 자동으로 관리하여 데이터 무결성을 확보한다. 이러한 절차는 코드의 오류와 가스 한도 문제, 타임스탬프 조작 및 무단 접근 위협에 대한 보안성을 충족한다. 또한 제안된 보안 자동화 모델은 표 3에 나타난 바와 같이, 스마트 컨트랙트의 설계, 배포, 트랜잭션 전송 및 실행 단계에서 STRIDE 요소별 보안 위협에 효과적으로 대응한다.

표 3. 보안 모델의 STRIDE 보안성 분석
Table 3. STRIDE security analysis of security models

STRIDE Element	Security Satisfaction Method	Satisf-action
Spoofing(S)	Implementation of authentication systems and multi-factor authentication to prevent phishing	○
Tampering(T)	Data verification based on programmed logic and automatic management of blockchain network timestamps and hash values	○
Repudiation(R)	Use of audit tools and testing systems for accurate transaction verification and prevention of repudiation	○
Information Disclosure(I)	Automatic identification of sensitive data and application of privacy policies, management of data access rights through ACL and RBAC	○
Denial of Service(D)	Calculation of gas limits before transaction processing and detection of abnormal transactions	○
Elevation of Privilege(E)	Establishment of an authentication server through Role-Based Access Control(RBAC) and application of off-chain techniques	○

V. 결 론

본 논문에서는 스마트 컨트랙트의 동작과정을 3단계로 분류하여 각 단계별 보안 위협을 분석하였으며, 이에 대한 보안 방안을 분석하였다. 이를 통하여 스마트 컨트랙트 동작과정의 각 단계별 보안 요구사항을 자동화하여 블록체인환경에서 안전한 스마트 컨트랙트의 보안 자동화 모델을 제안하였다. 이 모델은 스마트 컨트랙트 동작 과정의 각 단계별 다양한 보안 위협에 대응하며, 특히 스마트 컨트랙트 설계 및 배포단계에서의 코드 오류와 프라이버시 보호에 안정성을 확보하였다. 또한 트랜잭션의 전송에 있어서 인증 서버를 통해 멀티팩터 인증과 접근 제어를 통해 데이터 접근 권한을 명확히하고 무단 접근을 방지한다. 트랜잭션 실행 및 기록 단계는 검증된 트랜잭션만 실행하도록 하여 논리 오류, 가스 한도 문제, 타임스탬프 조작 등에 효과적으로 대처한다. 블록체인 데이터 변조와 불법적 접근에 대한 강력한 대응은 블록체인 시스템의 무결성과 신뢰성을 보장한다. 이러한 보안 접근법은 특히 블록체인 기술이 금융, 의료, 교육 등 다양한 분야에 적용되는 상황에서 매우 중요하다. 제안한 모델의 보안성 분석을 통해 모델의 각 단계별 보안 위협에 대한 보안성을 모두 만족하는 것으로 분석하였다. 향후 제안한 보안 모델을 실제 사례에 적용 및 구현하고, 보안 위협의 진화에 따라 모델을 지속적으로 개선해야 할 필요가 있다. 이를 통해 블록체인 기술의 보안 및 신뢰성을 더욱 강화하고, 광범위한 사용자에게 안전한 블록체인 환경을 제공할 수 있을 것으로 기대한다.

참고문헌

1] GeeksforGeeks. What Was the DAO Hack? [Internet]. Available: https://www.geeksforgeeks.org/what-was-the-dao-hack/.

[2] M. Swan, *Blockchain: Blueprint for a New Economy*, Sebastopol, CA: O'Reilly, 2015.

[3] G. Wood, Ethereum: A Secure Decentralised Generalised Transaction Ledger, Ethereum Project Yellow Paper, 2014.

[4] D. H. Sin and J. H. Lee, "Smart Contract Security for Pin Tech," *Korea Information Processing Society Review*, Vol. 22, No. 5, pp.54-62, September 2015.

[5] S. S. Kushwaha, S. Joshi, D. Singh, M. Kaur, and H.-N. Lee, "Systematic Review of Security Vulnerabilities in Ethereum Blockchain Smart Contract," *IEEE Access*, Vol. 10, pp. 6605-6621, 2021. https://doi.org/10.1109/ACCESS.2021.3140091

[6] I.-S. Kim, “Survey on Smart Contract Programming Languages,” *Electronics and Telecommunications Trends,* Vol. 35, No. 5, pp. 134-138, October 2020. http://dx.doi.org/10.22648/ETRI.2020.J.350512

[7] W. Mougayar, *The Business Blockchain: Promise, Practice, and Application of the Next Internet Technology*, Hoboken, NJ: Wiley, 2016.

[8] I. Allison, “Microsoft adds Ethereum language Solidity to Visual Studio,” International Business Times, March 2016.

[9] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, “Making Smart Contracts Smarter,” in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS ’16)*, Vienna, Austria, pp. 254-269, October 2016. https://doi.org/10.1145/2976749.2978309

[10] P. Tsankov, A. Dan, D. Drachsler-Cohen, A. Gervais, F. Bünzli, and M. Vechev, “Securify: Practical Security Analysis of Smart Contracts,” in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications (CCS ’18)*, Toronto, Canada, pp. 67-82, October 2018. https://doi.org/10.1145/3243734.3243780

[11] S. Kalra, S. Goel, M. Dhawan, and S. Sharma, “ZEUS: Analyzing Safety of Smart Contracts,” in *Proceedings of the 25th Network and Distributed System Security Symposium* (NDSS 2018), San Diego: CA, pp. 1-12, February 2018. https://doi.org/10.14722/ndss.2018.23082

[12] GitHub. Manticore: A Symbolic Execution Tool for Analysis of Smart Contracts and Binaries [Internet]. Available: https://github.com/trailofbits/manticore.

[13] GitHub. Mythril-Classic: An Open-Source Security Analysis Tool for Ethereum Smart Contracts [Internet]. Available: https://github.com/skylightcyber/mythril-classic.

[14] C. F. Torres, J. Schütte, and R. State, “Osiris: Hunting for Integer Bugs in Ethereum Smart Contracts,” in *Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC ’18)*, San Juan, Puerto Rico, pp. 664-676, December 2018. https://doi.org/10.1145/3274694.3274737

[15] M. Keller, “MP-SPDZ: A Versatile Framework for Multi-Party Computation,” in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS ’20)*, Online, pp. 1575-1590, November 2020. https://doi.org/10.1145/3372297.3417872

[16] C. Choi, Y.-J. Lim, Y.-J. Song, and J.-H. Lee, “Analysis of Off-Chain Solutions for Ethereum Scalability Issues,” in *Proceedings of KICS Winter Conference 2019*, Pyeongchang, pp. 208-209, January 2019.

[17] R. McRee, “Microsoft Threat Modeling Tool 2014: Identify & Mitigate,” *ISSA Journal*, Vol. 12, No. 5, pp. 39-42, May 2014.

조도은(Do-Eun Cho)

2008년 : 충북대학교 대학원 (공학박사)

2008년～현 재: 목원대학교 SW교양학부 교수

※관심분야 : 정보보호, 센서네트워크, 공학교육 등