핀테크를 위한 스마트 컨트랙트 보안

신다혜 · 이종협 (가천대학교)

목차

1. 서 론

비트코인과 같은 디지털화폐(암호화폐)가 최근 핀테크의 새로운 영역으로 각광을 받고 있는 것은 모바일이나 Internet of Things (IoT)와 같은 새로운 플랫폼과의 적합성 때문이다. 일차적으로는 서비스의 전 과정이 디지털 시스템 내에서 수행될 수 있다는 장점을 가지고 있지만 디지털 화폐 자체에 프로그램이 탐재 가능한 “programmable money”로 불리는 장점 때문이기도 하다. 다시 말하자면, 디지털 화폐는 단순히 주고 받는 데에서 끝나는 것이 아닌 주어진 역할에 맞는 역할을 하는 기능 또한 가지고 있는 셈이다. 이 모두를 가능하게 하는 것이 스마트 컨트랙트(Smart Contract)이다[1].

스마트 컨트랙트가 주목을 받는 이유는 기존의 금융 서비스들이 별도의 시스템 없이 디지털 화폐 환경 내부에 프로그램으로써 탑재될 수 있기 때문이다. 현재 금융 기관에서 수행하고 있는 일들이 사람들의 개입 없이 자동적으로 수행될 수 있어서, 관련 기술이 안정화 되고 난 이후에는 금융 기관의 역할마저도 대체할 수 있을 것으로 기대되고 있다. 새로운 IT 기술을 통하여 기존의 금융을 근본적으로 혁신하고자 하는 시대적 상황도 스마트 컨트랙트에 대한 중요성을 높이고 있다.

스마트 컨트랙트는 자율성을 가지면서 안전한 거래 시스템에 대한 요구에서 시작되었다. 미리 정해진 조건을 만족시키는 경우에 특정한 금융 거래가 이루어지도록 하는 것은 사실 모든 금융 서비스에서 가장 기본적인 일이라 할 수 있다. 이러한 일들을 정확하고 빠르게 수행하고자 현재의 금융은 어떤 산업에 못지않은 복잡한 IT 시스템을 이용하고 있으며, 혹시라도 이러한 일들이 정해진 대로 발생하지 않을 경우에 대비하여 법률 용어들로 가득한 계약서가 강제력을 보장한다. 스마트 컨트랙트는 이 모든 것을 디지털 시스템에 맡긴다. 주어진 계약 조건은 프로그램화 되어 자체적으로 수행되면서

동시에 예외 없이 실행되는 강제력을 가진다. 이러한 이유 때문에 스마트 컨트랙트는 암호학과 인터넷의 컴퓨터들에 의해서 신뢰성을 보장받는 암호화폐 시스템을 기반으로 구현되고 있다.

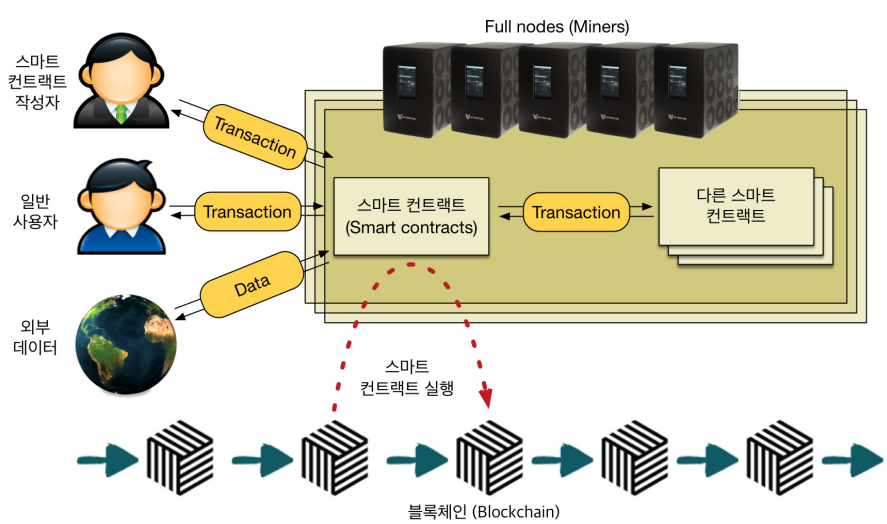
아직 스마트 컨트랙트 기술이 널리 사용되고 있는 기술은 아니지만 새로운 핀테크 금융 시대를 실현하기 위해서는 스마트 컨트랙트의 활성화가 필요하다. 본 고에서는 이러한 스마트 컨트랙트는 무엇이며 어떠한 가능성을 가지고 있는지, 또한 안전한 핀테크 환경을 만들기 위해서 스마트 컨트랙트와 관련된 보안의 문제점은 어떠한 것들이 있는지 알아보고자 한다.

2. 스마트 컨트랙트와 기반 기술

2.1 개요

Nick Szabo는 가장 단순한 스마트 컨트랙트의 형태를 자판기에 비유하였다[2]. 자판기가 주어진 금액과 선택이 있으면 조건에 맞는 제품을 제공을 하는 것이 마치 주어진 조건에 따라 동작하는 스마트 컨트랙트와 유사한 점이 있다는 것이다. 사실 스마트 컨트랙트는 거창한 혹은 딱딱한 이름과는 무관하게 암호화폐 시스템에서 동작하는 프로그램 코드의 조각에 불과하다. 하지만 암호화폐 시스템에 의해서 신뢰성을 보장 받으며, 화폐 시스템 내에서 동작하기 때문에 입력과 출력으로 금융 거래를 직접적으로 수행할 수 있다는 점에서 차이를 가진다.

기술적으로 정확하게 표현하자면 스마트 컨트랙트는 암호화폐의 블록체인에서 동작하는 프로그램이고, 블록체인에서 동작하기 때문에 실행(집행)의 신뢰성이 보장이 되는 방식이다. 따라서 스마트 컨트랙트를 설명하기 위해서는 기반이 되는 암호화폐의 블록체인에 대한 이해가 필요하다. 블록체인은 최근 암호화폐들이 공통적으로 사용하는 기반 기술로써 기본적인 목표는 인터넷을 통해 연결되어 있는 컴퓨터(노드)들에게 동일하고, 신뢰할 수 있으며, 수정 없이 추가만 가능한 문서(또는 장부)를 유지하게 하는 것이다. 특히 블록체인은 각각의 조건을 만족시키게 하기 위해서 hash chain과 작업증명(Proof of Work)과 같은 방식을 이용하고 있다. 암호화폐는 블록체인이 제공하는 신뢰할 수 있



(그림 1) 스마트 컨트랙트와 블록체인

는 장부에 발생하는 온갖 거래들을 기록하여 거래의 안전성과 정합성을 검증받는다. 중요한 점은 블록체인을 유지하고 있는 인터넷의 노드들에 의해서 발생한 거래의 정합성을 검증받는 과정이 거래 당사자가 제공한 프로그램 조각들을 모아 실행하는 과정이라는 것이다. 화폐와 연관되어 정당한 결과를 만들어내는 프로그램을 제공하는 것이 화폐의 소유권을 증명하는 방식으로 동작한다. 따라서 이러한 프로그램의 실행과정이 블록체인의 핵심 요소 중의 하나이며 동시에 암호화폐 안전성을 보장하는 핵심이다.

스마트 컨트랙트는 이러한 암호화폐의 프로그래밍 기능을 다양한 방면으로 확장하여 사용하는 방식을 의미한다. 따라서 스마트 컨트랙트란 블록체인에서 제공하는 프로그래밍 기능 자체를 의미할 수도 있지만, 단순한 거래 검증을 벗어난 주어진 조건에 따라 동작이 결정되는 정도의 복잡도를 가진 경우를 지칭하는 것이 일반적이다. 또한 블록체인은 시스템에 참여하는 컴퓨터들(정확하게 말하자면 full node라 불리는 적극 참여자 혹은 채굴자들)이 동일한 과정을 통해 동일한 검증된 장부를 유지하는 것이기 때문에 모두가 동일한 검증 과정을 각자 수행하는 셈이다. 따라서 스마트 컨트랙트 또한 각 노드에서 동일하게 실행이 된다. (그림 1)은 스마트 컨트랙트의 구성 및 동작에 대한 개념을 보여준다. 스마트 컨트랙트는 암호화폐 시스템 내부에 존재하며 transaction등을 통하여 사용자나 다른 스마트 컨트랙트들과 상호작용한다.

스마트 컨트랙트는 구현상의 어려움이 있을 뿐 화폐 시스템을 바탕으로 한 대부분의 작업을 수행할 수 있다. 정보 등록, 코인/쿠폰 발행에서부터 예측시장(prediction market) 구현, 파생상품 매매, 크라우드 펀딩, 인터넷 은행에 이르는 금융 시스템을 스마트 컨트랙트로 구축할 수 있다. 물론 이러한 기능들이 기존이 컴퓨터 프로그램에서도 가능한 일이었지만, 현재의 금융 시스템과 기존의 프로그램의 연동의 구조는 완전히 자율화되어 수행될 수 없으며 보안적으로도 안전성과 투명성을 보장할 수 없었다. 블록체인을 통하여 기존의 시스템에서는 해결하지 못했던 걸림돌을 제거하게 되어 스마트 컨트랙트가 실질적으로 사용될 수 있는 환경이 된 셈이다. 이러한 발전에 힘입어 스마트 컨트랙트를 극도로 활용하여 분산형 블록체인 위에서 존재하는 자율 조직인 DAO (Distributed Autonomous Organization) 형태로의 발전의 가능성도 타진되고 있다. 스마트 컨트랙트의 활용에 대한 요구가 높아지면서 스마트 컨트랙트의 기술적 부분에서 이러한 요구사항을 충족시킬 수 있는지에 대한 함께 이루어지고 있는 중이다.

2.2 스마트 컨트랙트 구현 기술 및 이더리움

스마트 컨트랙트는 기반이 되는 블록체인에 따라 기능과 구현 방식이 달라지게 된다. 현재 가장 많이 사용되고 있는 비트코인의 경우에는 transaction의 입력(input)과 출력(output)에 Script를 사용하도록 되어 있다. 비트코인의 거래가 이루어질 때 마다 (즉 transaction과 transaction이 연결될 때 마다) 입력과 출력의 Script가 실행된다. 일반적으로 거래에 담겨 있는 Script 프로그램들은 해당 거래의 주소에 대한 권한 증명(주소에 해당하는 공개키와 개인키를 이용한 서명)을 위한 전형적인 형태를 가지지만, 특수한 조건이 필요한 경우에는 이를 활용하여서 스마트 컨트랙트의 목적에 맞게 사용할 수 있다. Script 언어는 대부분의 프로그래밍 언어에서 제공하는 기능들을 제공하고는 있지만, 무한루프의 위험성 때문에 반복 기능은 제외되어 있다. 따라서 Turing-complete라 부를 수 없는 제한적인 기능을 가진다. 물론 모든 스마트 컨트랙트에 반복문이

필요한 것은 아니기 때문에 비트코인의 Script만으로도 많은 일들을 할 수 있지만, 더 완전한 기능을 제공하기 위하여 이더리움(Ethereum)이라 불리는 새로운 암호화폐 시스템이 나타나게 되었다.

이더리움은 차세대 블록체인 기술 중의 하나로 "Blockchain 2.0"이라 불리우는 암호화폐들 중의 대표주자라 할 수 있다[3]. 기본적으로는 비트코인과 같은 암호화폐 시스템이지만, 비트코인의 문제점들을 개선하고 Turing-complete한 프로그램 기능을 제공하는 새로운 플랫폼 구현을 목적으로 개발되었다. 이를 위해서 비트코인과는 전혀 다른 체계의 블록체인 기술을 개발하여 사용한다. 비트코인의 블록체인 기술이 분산 환경에서 안전한 디지털 화폐의 사용을 위한 신뢰성 제공에 초점을 맞추었다면 이더리움의 블록체인 기술은 스마트 컨트랙트와 같은 기능을 모두 내포할 수 있는 프로그래밍 능력에 초점을 맞추어 개발되었다. 따라서 비트코인과 같은 신뢰성 구조 위에 이더리움의 기능을 제공하기 위한 컴퓨팅 플랫폼 구조가 얹혀 있는 형태를 가지고 있다. 또한 이더리움에서는 스마트 컨트랙트가 transaction의 입력과 출력에 제한적으로 존재하는 것이 아니라 별도의 주소로써 존재할 수 있기 때문에 블록체인 시스템 내에서 스마트 컨트랙트가 transaction으로부터 독립되어 있는 개체로 동작할 수 있는 장점을 가진다. (그림 2)는 이더리움 시스템을 이용한 스마트 컨트랙트의 코드의 예이다.

이더리움이 흥미로운 점은 무한루프에 대한 걱정없이 반복 기능을 제공하기 위하여 프로그램 코드를 수행하는 동안 소진되는 gas의 개념을 가지고 있다는 것이다. 이 gas는 이더리움의 화폐로 지불되기 때문에 작업을 수행할 때마다 (적은 양이기는 하지만) 일정양의 금액을 소비해야 하는 셈이다. 따라서 보유하고 있는 gas의 양만큼만 한정적으로 실행될 수 있다. 또한 이더리움은 블록체인에서 실행되는 프로그램을 위한 중간단계를 가지고 있다. 비트코인의 경우에는 transaction에 담겨있는 Script 언어를 직접적으로 실행하는 형태이지만, 이더리움은 transaction에 포함되는 low level 코드를 별도로 정의하고 실제 스마트 컨트랙트 작성자들은 다른 프로그래밍 언어로 작성하고 (현재는 Python과 비슷한 문법 구조를 가지는 Serpent란 언어와 Lisp와 비슷한 LLL이란 언어가 일반적으로 많이 사용되고 있다.) 컴파일 과정을 통하여 실제 transaction에 담길 low level 코드로 변환되는 과정을 거치고 있다. 이 때문에 프로그래밍 언어와 실제 실행코드가 decoupling되어 스마트 컨트랙트 작성자가 하나의 언어에 종속되지 않는 장점을 가지고 있으며, LLVM과 같은 최신 컴파일러 플랫폼에서 추구하고 있는 공통 중간 언어(intermediate language)에 대한 최적화나 Just-In-Time(JIT)과 같은 기능의 적용이 쉽다는 이점을 가지고 있다.

```
def init():
        self.storage[msg.sender] = 10000
def code():
        to = msg.data[0]
        from = msg.sender
        value = msg.data[1]
        if self.storage[from] >= value:
                self.storage[from] = self.storage[from]  - value
                self.storage[to] = self.storage[to] + value
```

(그림 2) 이더리움에서의 스마트 컨트랙트의 예 (디지털 뱅킹)

3. 스마트 컨트랙트에서 발생할 수 있는 위험성

스마트 컨트랙트는 앞서 살펴본 것과 같이 암호화폐와 IoT 플랫폼의 발전에 필수적인 요소로써의 역할을 수행할 것으로 예상된다. 하지만 스마트 컨트랙트는 직접적으로 화폐를 다루거나 저장하는 프로그램이기 때문에 무엇보다도 신뢰할 수 있고 안전하게 실행되어야 한다. 잘 설계된 블록체인 기술에 의하여 스마트 컨트랙트의 실행 자체에 대한 공격이나 오류는 아직까지 크게 문제되지 않지만 오히려 정당한 문법을 가지는 스마트 컨트랙트들에서 발생 할 수 있는 위험요소들을 고려해야 한다. 1) 작성자의 실수로 인하여 논리적인 허점이 발생하거나, 2) 고려하지 못했던 corner case등에 의해서 스마트 컨트랙트가 제대로 동작하지 못하거나, 3) 익명성을 이용하여 범죄와 같은 악의적인 목적으로 사용되는 문제점들을 고민해 볼 수 있다.

3.1 정상적인 스마트 컨트랙트에 대한 공격

금융 거래를 발생시키거나 대상이 되는 스마트 컨트랙트의 특수성 때문에 일반적인 소프트웨어에 대한 공격에 비하여 즉각적인 이익을 얻을 수 있기 때문에 집중적 공격의 대상이 될 것으로 예상되고 있다. 실행되는 플랫폼이 다르기 때문에 기존의 일반적인 소프트웨어 취약점에 대한 공격들은 적용되지 않지만, 스마트 컨트랙트에서만 존재할 수 있는 버그를 찾아내기 위한 방어는 치밀하게 이루어져야 한다. 특히 스마트 컨트랙트가 모든 노드에서 동일하게 수행되어야 한다는 특징 때문에 오히려 실제 실행코드는 누구라도 손쉽게 얻을 수 있고 필요하다면 추가적인 disassemble 과정을 통하여 스마트 컨트랙트 원본을 알 수 있기 때문에 더욱 조심해야 한다.

최근 메릴랜드 대학교의 Kevin Delmolino 등의 연구원은 [4]에서 스마트 컨트랙트를 작성하는 과정에서 고려해야하는 주의사항들을 다음과 같이 정리하였다.

1) State 설계의 오류: 스마트 컨트랙트에서 발생할 수 있는 논리적인 문제점이다. 스마트 컨트랙트가 생각하지 못하던 경우에 빠지게 되어 더 이상 실행이 진행되지 않는 경우를 의미한다. 스마트 컨트랙트에 단순화하면 외부의 입력에 의해 transition이 일어나는 state machine으로 모델링 할 수 있는데, 스마트 컨트랙트의 가능한 모든 transition과 state를 제대로 고려하지 않은 경우에 발생할 수 있다. 해커와 같은 소프트웨어 보안의 공격자들이 소프트웨어에 Logic bomb이라 불리는 조작된 입력을 주입하여 이상 상태를 유발하고자 하는 것과 같은 맥락의 문제점이다. 결국 스마트 컨트랙트에서 발생할 수 있는 모든 실행 path를 방문하며 발생할 수 있는 문제를 확인하는 절차가 필요하다.

2) 암호학 사용 과정에서의 실수: 스마트 컨트랙트의 기반이 되는 암호화폐는 분산형 시스템이며 인터넷을 통하여 거래와 같은 새로운 메시지 전달이 발생하기 때문에 네트워크 보안 분야에서의 암호 프로토콜을 설계하는 것과 비슷한 방식으로 스마트 컨트랙트에 (transaction에 해당하는) 메시지 전달 프로토콜을 설계할 수 있다. 하지만 암호화폐는 현재의 일반 인터넷 프로토콜들과는 다르게 블록체인 내의 transaction을 모두가 검증하기 위하여 메시지의 기밀성이 제한적으로 지켜지게 된다. 따라서 기존의 기법을 그대로 적용하는 데에는 한계가 있으며 여기서 발생하는 불일치(mismatch) 때문에 의도치 않게 스마트 컨트랙트 작성과정에서 메시지의 정보를 노출하는 실수를 할 수 있다.

3) 자체적 인센티브 구조의 부재: 암호화폐의 분산형 시스템에서는 특정할 수 없는 상대방에 대한

100% 신뢰를 가질 수 없다. 블록체인에서는 이러한 이유로 시스템에 반하는 행동을 하지 않는 경우에는 보상을 얻을 수 있는 인센티브 구조를 가짐으로써 해결하고 있다. 불특정 다수를 고려해야하는 것은 블록체인 위에서 동작하는 스마트 컨트랙트도 마찬가지이다. 따라서 스마트 컨트랙트를 설계하는 과정에서 자체적으로 참여자가 주어진 rule에 적극적으로 동참할 수 있도록 하는 인센티브 구조를 고려하여 같이 넣어야 한다. 또한 스마트 컨트랙트를 동작시키거나 호출하는 과정에서 발생하는 비용에 대한 보상을 위해서도 부가적인 인센티브 구조가 검토되지 않은 경우에는 의도와 다른 형태로 악용되거나 공격을 받을 수 있다.

4) 다중 스마트 컨트랙트 사용 과정에서의 문제: 이더리움과 같이 스마트 컨트랙트의 독립성이 높아진 경우에는 스마트 컨트랙트들 사이의 거래나 메시지 전달이 더 활발하게 이루어 질 것이다. 이때 스마트 컨트랙트 내부적 문제점 외에 기반 플랫폼의 특성에 따라 발생할 수 있는 문제도 함께 고려되어야 한다. 예를 들어, 이더리움의 경우에는 다른 스마트 컨트랙트에게 보낼 수 있는 메시지의 중첩 단계가 제한되어 있어 중첩된 메시지 전달이 반복적으로 발생하는 상황에서는 스마트 컨트랙트가 제 역할을 수행하지 못하는 취약점을 가지고 있다.

3.2 Criminal Smart Contracts (CSC)

정상적이고 합법적인 스마트 컨트랙트에 대한 공격은 작성자의 의도와는 다르게 실행되도록 하는 것이 일반적이었다면 이와는 다른 접근으로써 암호화폐의 익명성을 바탕으로 하여 불법적인 일에 스마트 컨트랙트를 이용하고자 하는 CSC들의 문제가 제기 되고 있다[5]. 기존의 CSC등의 경우에는 일종의 프로토타입이나 개념적인 형태로 제시되어 왔다면, 오히려 스마트 컨트랙트의 신뢰성이 높아짐에 따라서 안정적으로 동작하는 CSC가 출현할 수 있게 되어 실제적으로 보안의 위협으로 발전할 가능성이 높아지게 되었다.

대표적인 CSC 형태는 습득한 비밀을 누설하는 것에 대해 보상을 받는 public leakage 들이다. Darkleaks[6]는 이러한 서비스를 비트코인 위에서 구현된 시스템 중 하나이다. 누출하고자 하는 정보가 있는 계약자는 자신이 받고자 하는 목표액을 설정하고, 해당 정보를 여러 조각으로 나누어 암호화하여 Darkleaks 시스템에 제공한다. 그러면 시스템에서는 랜덤하게 조각들의 일부를 선택하여 복호화를 요청하고, 복호화 된 조각들을 샘플로써 공개하여 모든 사람들이 열람할 수 있게 한다. 크라우드펀딩과 같이 누출한 샘플을 바탕으로 외부의 참여자들은 해당 정보가 충분한 가치가 있는지를 판단하고 전체 정보를 얻기 위하여 일정 비용을 지불할지를 결정하여 알려준다. 참여한 금액의 누적 액이 목표액을 초과하는 경우에는 모든 조각들에 대한 암호키를 제공함으로써 해당 정보를 모두 누출하게 된다. 계약자가 모든 조각에 대한 암호키를 제공하지 않고 금액만 챙겨가는 경우를 방지하기 위하여 Darkleaks 시스템에서는 암호키를 모금액이 들어있는 비트코인 주소들의 공개키의 hash값으로 설정함으로써 돈을 가져가기 위해서 생성하는 transaction 자체에서 암호키의 정보가 드러나게 구성하였다. 즉 목표액의 비트코인을 가져가는 행위 자체에서 키가 드러나는 형식이 된다. [5]에서는 이러한 Darkleaks 시스템이 기반하고 있는 비트코인에서 구현될 수 있는 스마트 컨트랙트의 한계점 때문에 여러 가지 공격이 가능하다는 것을 보이며 이더리움의 스마트 컨트랙트를 사용하는 강화된 버전을 제시하고 있다.

Darkleaks와 같은 시스템은 정보를 폭로하는 계약자나 이를 얻고자 하는 참여자들에 대한 익명성이 지켜지며, 완전히 이더리움의 스마트 컨트랙트

로 구현될 경우에는 관리하는 주체없이 독립적이고 자율적으로 동작할 수 있게 된다. 규제받지 않으면서 동작한다는 점에서 순기능을 가지고 있기도 하지만, 불법적인 용도로 생성된 경우에는 어떻게 탐지하고 규제해야 하는지에 대한 고려가 필요하다.

4. 안전한 스마트 컨트랙트를 위한 보안 기법

스마트 컨트랙트에 의해서 발생할 수 있는 위험성을 방지하거나 제거하기 위해서는 안전한 스마트 컨트랙트를 구성하고 악의적인 스마트 컨트랙트를 탐지해내는 과정이 필요하다. 암호화폐 플랫폼이라는 특수성이 있지만, 스마트 컨트랙트는 스크립트 언어에서 시작한 프로그램이라는 점에서 기존의 소프트웨어 보안에서 사용하는 보안 기법들을 적용할 수 있다. 비록 스마트 컨트랙트의 기능과 구현 방법이 계속 발전하고 있는 상황이기는 하지만, 근본적인 보안의 위협은 동일하다. 따라서 각각의 문제점에 대한 대응으로써 다음과 같은 접근 방법으로 보안을 강화할 수 있다.

4.1 프로그램 분석 기법을 이용한 문제점 탐지

소프트웨어 보안에서 사용하는 프로그램 분석 기법들을 스마트 컨트랙트에 적용하여 발생할 수 있는 문제점을 탐지한다. 스마트 컨트랙트는 거래가 발생하는 경우에 실행되기 때문에 event-driven 형태의 프로그램과 비슷하면서도 프로그램이 오랜 시간에 걸쳐 다른 거래들과도 연동하여 수행될 수 있기 때문에 기존의 프로그램에 대한 모델링과는 차이점을 가지고 있다. 또한 스마트 컨트랙트는 점점 더 복잡한 형태로 발전하고 있지만 일반적인 소프트웨어에 비하면 구조가 단순하기 때문에 정적 분석(static analysis)를 적용하기에 알맞다. 하지만 스마트 컨트랙트가 내부적으로는 state들을 가지고 있으며 실행 당시의 state가 블록체인에 저장되어 있기 때문에 이를 반영할 수 있는 방법이 고려되어야 한다. 최근에는 [5,7,8]와 같은 연구에서 스마트 컨트랙트에 대해 정형 기법(formal methods)나 timed automata를 이용하여 모델링하는 방법들이 연구되었지만, abstract interpretation등을 활용하는 전통적인 정적 분석 기법들을 통한 탐지 기법의 연구도 기대해 볼 수 있다.

4.2 안전한 스마트 컨트랙트 작성 지원

작성된 스마트 컨트랙트를 분석하기 이전에 스마트 컨트랙트 자체에 안전성을 확인할 수 있는 장치들을 포함할 수 있다. 일반적인 프로그램 언어에서는 프로그램의 수행 중간에 이상 상태를 확인하기 위해 주어진 조건을 확인하는 assertion(assert 문)을 이용한다. 스마트 컨트랙트에서도 실행과정에서 변경되지 말아야 하는 보안 조건들(i.e., invariant)을 설정하고 실행 과정 중에 주기적으로 확인하는 방식을 도입할 수 있다. 다만 단순한 assertion이라도 스마트 컨트랙트에 적용되기 위해서는 고려해야할 사항이 생긴다. 특히 일반 프로그램에서는 주어진 조건에 위배하는 경우에 실행을 중단하면 되지만, 스마트 컨트랙트의 경우에는 단순히 중단하는 것이 아니라 이전 state로 회귀하는 롤백(roll back)형태의 과정이 모든 노드에서 동일하게 이루어질 수 있도록 하는 등의 이슈가 있어 어려움이 있다. 또한 이러한 보안 조건들을 사용자의 의도에 해당하는 조건들로 발전하여 스마트 컨트랙트의 정확한 specification을 정의하고 표현하는 방법들도 개발되고 있다.

4.3 악의적 스마트 컨트랙트의 탐지 및 차단

CSC와 같이 악의적인 스마트 컨트랙트를 방지하기 위해서는 적절한 대응을 위해서는 CSC를 탐지하고 차단하는 방법들이 필요하다. CSC의 탐지는 기존 프로그램의 분석의 behavior analysis와 유사한 과정을 통하여 수행할 수 있다. 아직까지 스마트 컨트랙트는 각각 코드 자체에 대한 난독화는 잘 이루어지지 않고 있다. 따라서 실제 동작 과정을 쉽게 드러내지 않기 위해서 하나의 스마트 컨트랙트를 다수의 스마트 컨트랙트로 쪼개어 수행하도록 하는 방식으로 난독화가 발전될 것이다. 이 때문에 CSC에 대한 정적 분석 기반의 행동 분석에 어려움이 따르게 된다. 하지만 스마트 컨트랙트는 모든 노드에서 직접 실행되어야 하므로 실행과정에서 이상 상태를 찾아내는 모듈을 노드에 탐재하는 동적 분석(dynamic analysis)기법의 적용을 고려해야 한다. 이렇게 CSC가 탐지되더라도 이를 차단하는 과정에서 또 다른 이슈가 발생한다. 암호화폐의 블록체인은 기본적으로 중앙의 관리를 받지 않는 분산형 구조이기 때문에 악의적인 스마트 컨트랙트를 탐지한다고 하더라도 차단을 결정하는 관리 주체가 없는 셈이다. 따라서 악의적인 스마트 컨트랙트를 명확하게 정의하고 차단하는 기능을 모든 노드에서 동일하게 적용되도록 해야 하는데, 이미 스마트 컨트랙트 자체를 수행하는데 부하를 가지고 있는 노드들에게 탐지 및 차단을 위한 작업을 부가하여야 하며, 지속적으로 우회책을 사용할 CSC들에 대해서 모든 노드가 기민하게 대응하도록 조정하는 것이 쉽지 않아 지속적인 연구가 필요하다.

5. 결 론

디지털 화폐의 진정한 가능성은 스마트 컨트랙트의 구현과 활용을 통해서 열리게 된다. 암호화폐가 단순히 화폐의 대체품이 아닌 새로운 금융의 시대를 열기 위해서는 스마트 컨트랙트의 적극적 도입과 활용이 중요하다. 하지만, 스마트 컨트랙트가 가지는 중요성이나 역할에 비례하여 잘못 사용된 스마트 컨트랙트가 초래하는 위험성도 크다 할 수 있다. 스마트 컨트랙트는 아직 널리 사용되지 못하고 구현 방식에 대한 변화가 많은 성장기에 있는 기술이기는 하지만, 발전 과정에서 발생할 수 있는 보안의 문제점이나 악영향에 대한 연구와 기술 개발을 통하여 신뢰성과 안전성에 대한 보장을 스마트 컨트랙트 생태계 자체 내에 포함시킬 수 있어야 한다. 특히 디지털 금융의 미래를 준비하며 기존의 것에서 머무르지 않는 새로운 핀테크만의 기술과 서비스에 대한 요구가 활발한 요즘이 안전한 스마트 컨트랙트의 개발 및 보안 기술에 대한 관심을 기울일 때라 할 수 있다.

참 고 문 헌

[1] Smart contract, http://https://en.wikipedia.org/wiki/Smart_contract

[2] Nick Szabo, "The idea of smart contracts," http://szabo.best.vwh.net/smart_contracts_idea.html

[3] Ethereum, https://www.ethereum.org

[4] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab," IACR Cryptology ePrint Archive

[5] A. Juels, A. Kosba, and E. Shi., "The Ring of Gyges: Using Smart Contracts for Crime." http://arijuels.com, (in submission.) 2015.

[6] Darkleaks, https://github.com/darkwallet/darkleaks

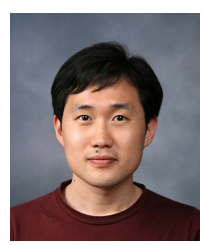
[7] M. Andrychowicz, S. Dziembowski, and D. Malinowski, "Modeling Bitcoin Contracts by Timed Automata." arXiv.org.
[8] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts," IACR Cryptology ePrint Archive

저 자 약 력

신 다 혜

이메일: sdhdonna@naver.com

- 2014년 3월~현재 가천대학교 금융수학과 재학
- 관심분야 : 금융 보안, 금융 공학



이 종 협

이메일: jonghyup@gachon.ac.kr

- 2009년 8월 연세대학교 컴퓨터과학과 졸업 (공학박사)
- 2009년 9월~2012년 2월 Carnegie Mellon, CyLab 연구소 (박사후연구원)
- 2012년 3월~2015년 2월 한국교통대학교 소프트웨어학과 조교수
- 2015년 3월~현재 가천대학교 금융수학과 조교수
- 관심분야 : 금융 보안, 소프트웨어 보안