공학석사 학위논문

마이데이터를 활용한 블록체인 기반 저작권 데이터 관리 및 공유 플랫폼

2020년 2월

부경대학교 대학원

정보보호학 협동과정

김 혜 빈

공학석사 학위논문

마이데이터를 활용한 블록체인 기반 저작권 데이터 관리 및 공유 플랫폼

지도교수 신 상 욱

이 논문을 공학석사 학위논문으로 제출함.

2020년 2월

부 경 대 학 교 대 학 원

정보보호학 협동과정

김 혜 빈

김혜빈의 공학석사 학위논문을 인준함

2020년 2월

주심　이학박사 이 경 현　(인)

위원　이학박사 신　원　(인)

위원　이학박사 신 상 욱　(인)

차례

그림 차례

표 차례

Blockchain-based Copyright Data Management and Sharing Platform using the MyData Concept

Hyebin Kim

Interdisciplinary Program of Information Security,
The Graduate School, Pukyong National University

Abstract

An increasing number of copyright holders, consumers, and copyrighted works services platforms have stimulated the copyright data ecosystem. As a result, rights relationships related to the sharing, use, and settlement of works have become more complex. The copyright of the copyright holder's own work needs to be protected. Until recently, however, many cases of copyright infringement occurred by consumers or distribution brokers. This has resulted in copyright owners not being paid properly, or unauthorized use of copyright by unauthorized users. Therefore, the importance of the integrity, transparency, and reliability of copyright information registered in the system and the copyrighted data and the usage records stored on the service platform has emerged in order to protect the copyright of copyright holders.

Blockchain is a P2P network-based distributed ledger technology, and related works have been studied as it is expected to be able to compensate for some of the shortcomings of the existing copyright management model that lacks the features discussed. Blockchain allows the design of fully decentralized C2C models that theoretically do not require intermediaries. In practice, however, copyright management involves a variety of actors, especially for service platforms, which are necessary for areas such as user convenience. For copyright-oriented copyright management in the Blockchain network formed by a kind

of intermediaries such as a service platform, MyData, a private-centric data integration management, and control model, can be applied.

This thesis first analyzes the existing copyright management and copyright sharing model and discusses the limitations. It then proposes a consortium Blockchain-based copyright management model in which the service platform participates as a node, and discusses how to combine the My Data concept with Blockchain and smart contracts. Also, Blockchain-based CP-ABE is introduced and applied to the proposed model as a way for users to define access policies and store copyright data in encrypted form on the storage of the online service providers (OSP).

Compared with the existing copyright management model, the proposed model allows the copyright holder to focus on copyright registration, license content design, and sharing, as the data subject. And it is expected to be able to transparently manage the usage records and the basis for the settlement of the copyrighted data that are shared and used on each platform.

I. 서론

1. 연구배경

저작물(창작물)은 사람의 사상이나 감정을 일정한 형식으로 창작하여, 이를 다른 사람이 느끼고 깨달을 수 있도록 표현한 것이다[1]. 그리고 저작권(Copyright)은 창작물을 만든 이가 본인의 저작물에 대해 가지는 배타적인 법적 권리로써, 저작물을 보호한다[2]. 저작물을 서비스하는 플랫폼의 증가와 함께 다양한 종류의 수 많은 창작물들이 저작권자와 소비자 사이에서 거래 및 공유되어왔다. 이에 따라 서비스 플랫폼과 저작권자 그리고 소비자간에는 복잡한 이용 및 권리 관계가 존재한다. 이를 명확하게 하기 위하여 저작물의 저장 및 관리 그리고 저작물 이용 기록에 대한 신뢰성 보장의 중요성이 대두되었다.

기존의 저작권 및 저작물 공유 시스템은 저작권자가 서비스 플랫폼에 저작물 데이터를 등록하면, 플랫폼이 그것들을 관리하며 필요한 소비자에게 게시하거나 판매한다. 데이터 이용 기록을 근거로 하여 판매 수익을 정산하고, 수수료를 제한 후 저작권자에게 저작권료를 지급하는 형태이다. 다시 말해 서비스 플랫폼이 중앙의 데이터 공유 중개자 역할을 수행하는 셈이다.

문제는 이러한 시스템은 클라이언트-서버(Client-Server) 구조의 중앙 집중형 시스템의 단점을 상속한다. 저작권자는 자신의 데이터를 공유하는 데 있어 서비스 플랫폼의 관련 정책에 의존하며, 서비스 플랫폼이 제공하는 이용 기록을 신뢰할 수 밖에 없다. 그러나 서비스 플랫폼의 부당한 이익 취득을 위한 이용 기록 위•변조 사건이 다수 발생해왔다[3]. 또한 소비자들의 저작물 무단 사용 및 재 배포 문제도 끊임없이 문제점으로 제기되고 있다. 이러한 저작권 침해사고들은 저작권자로 하여금 창작의 의지를 위축시키고 이로 인해 이용할 수 있는 저작물이 줄어들게 되면

저작물 산업계가 원활하지 못하게 될 수 있다.

최근 들어 4차 산업혁명 기술 중 하나인 블록체인(Blockchain)을 저작권 관리 모델에 적용함으로써 [4]에서와 같이 이용 기록의 신뢰성 및 투명성을 보장하여 저작권자 중심의 저작물 공유를 하는 시스템에 대한 많은 연구가 진행되어 왔다. 이는 블록체인이 P2P(Peer-to-Peer) 네트워크라는 특징을 통해 서비스 플랫폼과 같이 중개자가 존재하지 않는 C2C(Consumer to Consumer)모델을 구현해 낼 수 있을 것으로 기대되었다. 이는 저작물 생태계의 또 다른 패러다임을 불러 왔음은 분명하다. 하지만 개인이 온전히 저작권을 주장하거나, 서비스 플랫폼을 완전히 배제시키는 것은 실질적으로 불가능하다. 또한 소비자의 이용 기록 위반에 대해서 완전히 파악하지 못하는 점도 존재한다.

따라서 서비스 플랫폼들을 배제시키지 않은 현실적인 모델에서 중앙 정책에 의존하지 않고 저작권자 중심으로 저작물을 공유할 수 있는 시스템에 대한 연구가 필요하다. 본 논문에서는 GDPR(General Data Protection Regulation)에 의거한 개인 데이터 관리 정책인 마이데이터(MyData)[6]를 이용하여 서비스 플랫폼을 호스팅하는 조직들로 구성된 블록체인을 기반으로 하는 저작권자 중심의 저작권 데이터 관리 및 공유 플랫폼을 제안하고 설계하고자 한다.

2. 연구 내용 및 구성

본 논문에서는 기존 저작권 관리 및 저작물 공유 시스템과 그의 문제점에 대해 분석한다. 이를 보완하기 위한 방안으로 마이데이터와 블록체인에 대해 논의한다. 마이데이터는 개인이 주체가 되어 자신과 관련된 모든 데이터, 즉 개인 데이터(Personal Data)를 본인이 온전히 관리할 수 있다는 개념으로써, 저작물에 대한 이용 동의(Consent)를 기반으로 한다. 그리고 블록체인을 이용하면 서비스 플랫폼 간의 네트워크를 설계하여 그들간의 데이터 관리를 수행할 수 있다. 블록체인은 현재 운영되고 있는 중앙 집중형 시스템을 탈 중앙형(decentralized) 시스템으로

전환하여 시스템 사용자들에게 데이터 무결성 및 투명성을 제공하는 역할을 할 수 있다. 또한 함께 사용되는 스마트 계약은 시스템에서 사용되고 있는 모든 거래 및 공유 프로세스를 블록체인과 결합하여 사용함으로서 대체할 수 있다 그리고 효과적인 데이터 공유를 위한 방법으로 암호문 정책 속성 기반 암호화(CP-ABE, Ciphertext Policy-Attribute based Encryption)를 제안 모델에 적용한다.

그리고 제안 시스템의 참여 구성 요소와 사용될 수 있는 스마트 계약 및 사용 시나리오에 대해 제안하고, 이를 바탕으로 하이퍼레저 패브릭 네트워크와 체인코드를 설계한다.

본 논문의 구성은 다음과 같다. 2 장에서 제안 시스템을 설계하기 위한 관련 연구로 마이데이터와 블록체인, 그리고 분산 권한 속성 기반 암호화에 대해 분석한다. 그리고 3 장에서 마이데이터 개념을 도입한 블록체인 기반 저작권 관리 모델을 제안한다. 3 장에서 기술한 제안 모델의 구성요소 및 작업 수행 방법을 토대로 4 장에서는 하이퍼레저 패브릭(Hyperledger Fabric)과 그에 사용하는 스마트 계약인 체인코드를 이용한 제안 모델의 구현 내용을 설명하고 사용 기법에 대하여 분석한다. 마지막으로 5 장에서 결론을 짓고 마무리한다.

II. 관련 연구

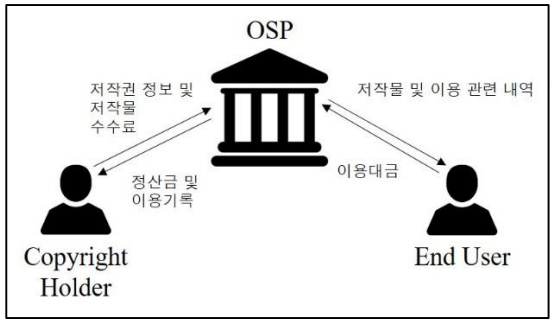
본 장에서는 기존의 저작권 및 저작물 데이터 공유 시스템에 대해 논한 후 유통과정에서 발생하는 문제점을 분석한다. 그 후 문제점을 보완할 수 있는 새로운 저작권 관리 시스템에 필요한 개념인 마이데이터(MyData)와 블록체인, 그리고 분산 권한 속성 기반 암호화(Multi-Authority CP-ABE)에 대해 논한다.

1. 기존 저작권 관리시스템과 마이데이터

가. 기존 저작권 관리 및 저작물 공유 시스템

기존 저작권 관리 시스템에서 저작권자들은 자신의 저작권을 본인이 직접 관리하거나 신탁관리단체에 위탁하여 관리할 수 있다[5]. 저작권 데이터는 모두 저작권 등록시스템을 통해 등록할 수 있으며 중앙 데이터베이스에 모든 저작권데이터가 저장되어 있다. 일반 사용자들은 임의의 저작물에 대한 저작권 데이터를 저작권 관리 시스템에 접속하여 요청할 수 있다.

저작권자는 유통사업자의 플랫폼을 통해 자신의 저작물을 게시하며, 소비자는 플랫폼에 게시된 저작물을 공유할 수 있다. [그림 1]은 현재 운영되고 있는 저작물 유통플랫폼에서의 유통과정을 나타낸다. 다음은 참여 구성요소에 대해 자세하게 논한다.



[그림 1] 기존 저작권 관리 및 공유 모델

● 저작권자(Copyright Holder): 저작권자는 저작물 데이터를 생성한 원작자로서 자신의 저작물에 대해 공유 및 거래와 같은 곳에서 권리를 행사 할 수 있다. 저작물을 생성하여 배포하면 저작권자의 권리가 인정되지만, 이를 좀 더 쉽게 증명하기 위하여 저작권을 데이터로 등록할 수 있다.

● 소비자(Consumer): 저작물을 최종적으로 소비하는 일반 사용자 모두를 일컫는다. 이들에게는 거래 및 공유 과정에서 저작권 법을 지켜야할 의무가 있다.

● 온라인 서비스 사업자 (Online Service Provider, OSP): 저작권자가 생성한 데이터는 그것을 재생, 게시, 거래, 공유할 플랫폼이 필요하다. 온라인 사업자들은 위와 같은 역할을 수행하는 플랫폼을 저작권자와 소비자에게 제공함으로써 수익을 창출한다. 현재 잘 알려진 유통 플랫폼으로는 유튜브(Youtube), 사운드클라우드(Soundcloud) 그리고 핀터레스트(Pinterests)등이 존재한다.

저작권자는 유통플랫폼을 통해 자신의 저작물을 게시하고 소비자들과 공유함으로써 자신의 이름을 알리고, 더 활발한 저작 활동을 하게 된다. 소비자들은 플랫폼을 통해 거래 또는 구독하고자하는 저작물을 손쉽게 찾을 수 있으며 유통사업자들은 저작권자와 소비자로부터 받은 피드백을 통해 더 나은 기능을 제공하는 플랫폼으로 개선하려고 한다. 또한 OSP 는 저작권자와 소비자사이에 발생한 거래 수수료나, 플랫폼 대여료 또는 광고료 등을 통하여 수익을 창출할 수 있다.

현재의 저작권 관리 시스템은 저작권자와 소비자가 저작권 관리를 하며 손쉽게 저작물을 공유할 수 있다는 장점이 있는 반면에, 다음과 같은 단점이 존재한다.

● OSP 의 저작물 이용 기록 위 변조: 저작권 관리 및 저작물 공유 시스템에서의 정산은 서비스 플랫폼에서 기록한 이용 기록을 기반으로 이루어지는 경우가 대부분이다. OSP 와 저작권자는 저작권 데이터에 명시된 정산 규칙을 준수해야 한다. 그러나 실제로 OSP 측에서 이용 기록을 조작하여 저작권자가 받을

이익을 중간에 가로채는 사건이 다수 발생하였다. 이는 저작권과 관련된 데이터가 중앙 데이터베이스에 저장이 되어있어 DB 및 서버 관리를 하는 OSP 의 내부 직원 외에는 실시간으로 투명하게 확인 하기 힘든 점을 이용한 것이다.

- 소비자의 저작권 이용허락 미 준수: 소비자들이 저작권 이용허락 규칙을 무시하고 저작물을 사용함으로써 저작권 침해가 발생할 수 있다. 해당 사례로써, 원 저작권자를 밝히지 않은 상태로 타 유통 플랫폼에 업로드 하거나 저작물을 2 차 가공하는 등의 경우가 있다. 워터마크(Watermark)등으로 저작권자를 명시할 수 있지만 이는 근본적인 해결책이 되지 않는다. 이러한 방식으로 저작권자 모르게 타 유통 플랫폼에 재 배포된 저작물은 이용 허락을 위반한 사용자가 수익이나 명성을 얻는 등 원 저작권자의 권리가 훼손될 수 있다.

나. 마이데이터

마이데이터(MyData)는 정보의 주체가 되는 개인이 본인 정보를 적극적으로 제어하여 이를 이용한 신용관리, 자산관리, 건강관리등 생활에 주도적으로 활용할 수 있는 일련의 과정들을 지칭한다[6]. 이는 분산된 본인의 데이터를 한 곳에 모아 관리 할 수 있도록 한다.

마이데이터의 핵심은 데이터 소유주 중심으로 데이터 관리 및 제어를 수행한다는 점이다. 현재 데이터 관리가 데이터를 가지고 있는 조직을 중심으로 하여 이용동의가 이루어진다. 반면 마이데이터 모델을 적용하게 되면 데이터를 사용하고자 하는 사용자가 데이터를 소유하고 있는 플랫폼에 데이터 사용 및 제공 요청을 할 때 해당 데이터를 소유 및 서비스하는 플랫폼 사업자가 소유자의 이용 동의를 요청한다. 소유자의 동의가 이루어지고 나면 데이터 공유가 가능하다[7]. 이러한 방식을 통하여 정보 주체는 자신이 동의한 수준에 따라 데이터를 이동하거나 처리할 수 있다.가장 대표적으로 알려져 있는 모델로는 핀란드 교통 통신부가 내놓은 MyData 모델로써 마이데이터 계정 정보를 기반으로 정보 제공의

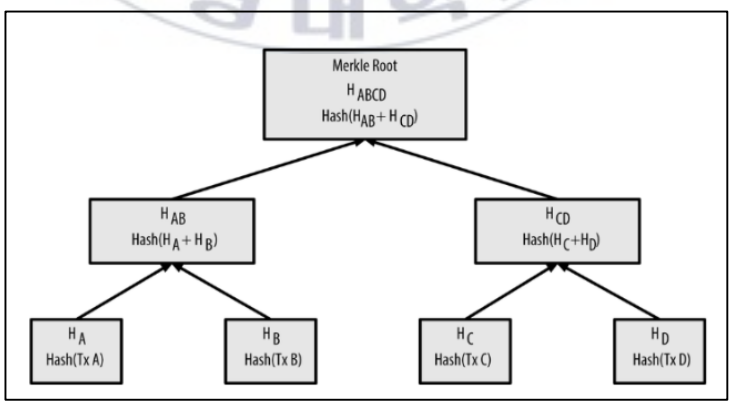
동의서를 관리한다. 해당 계정정보는 마이데이터 운영자(Operator)가 관리한다. 그 외에 다른 마이데이터 모델로는 영국의 MiData, 그리고 미국의 스마트 공시(Smart Disclosure) 모델등이 존재한다[8].

2. 블록체인과 스마트 계약

가. 블록체인

블록체인은 P2P 네트워크를 기반으로 하는 탈중앙화 분산 원장기술(Decentralized Distributed Ledger Technology)이다[9]. 2008 년 사토시 나카모토(Satoshi Nakamoto)가 제안한 비트코인 거래를 위한 기반 기술로써 처음 등장하였다. 블록체인의 핵심은 P2P 네트워크에 있는 노드가 작업 증명(Proof Of Work, PoW)과 같은 합의 메커니즘(Consensus Mechanism)으로 분산 장부를 동일하게 유지 및 관리한다는 것이다[10]. 이는 기존의 클라이언트-서버 구조에서는 서버가 TTP(Trusted Third Party)로서 저장된 데이터의 신뢰성을 보장하였으나, 블록체인 네트워크는 TTP 가 존재하지 않는 탈 중앙화 모델이기 때문에 다른 방법으로 저장된 데이터의 신뢰성을 보장하는 방법이다.

블록체인의 데이터를 네트워크에 있는 노드라면 누구나 열람할 수 있도록 투명하게 공개하면 데이터를 임의적으로 위•변조 하지 못한다. .



[그림 2] 블록체인의 머클 트리[11]

블록체인은 앞서 논한 투명성(transparency)을 제외하고도 두 가지 대표적인 성질을 지니고있다. 첫번째로 무결성(integrity)의 보장이다. 블록 하나는 수 많은 트랜잭션을 포함하고 있으며, 해당 트랜잭션들은 머클 트리(Merkle Tree)의 형태를 이루고있다. 머클 트리의 최상위 값(Root value)는 해당 블록에 포함된 트랜잭션의 요약본이라고 할 수 있으며, 블록 헤더에 포함된다. 머클 트리는 각 내부 노드내의 데이터를 두번 해시하여 같은 토폴로지의 노드와 더한다음 해싱을 반복한 값을 상위노드에 저장하는 작업을 최종적으로 루트 노드값이 계산될 때 까지 재귀적으로 수행한 결과 값이다. 암호학적 해시함수를 통해 계산되기 때문에, 트랜잭션값이 1 비트라도 변경이되면, 최종적으로 머클 루트값이 변경된다. 따라서 머클 루트를 포함하고 있는 블록 헤더 해시 값이 변경되고, 이는 다음 블록이 이전 블록의 해시(Previous hash)값으로 저장하고 있기 때문에, 뒤 이은 모든 블록 내용이 변경된다.

또 다른 특성은 불변성을 보장하는 것이다. 임의의 블록이 한번 연결되고 나서 안에 포함된 데이터 값을 변경하려고 하면, 해당 블록 이후로 연결되는 블록과 내부에 포함된 트랜잭션 값을 수정해야하기 때문에, 실제로 임의의 블록내에 있는 값을 수정하는 일은 불가능하다고 알려져있다. 따라서 한번 기록된 데이터의 불변성을 보장한다. 데이터가 한번 기록되면 수정하기가 쉽지 않기 때문에, 특정 트랜잭션과 관련된 이전 트랜잭션들과 결과값들이 기록 되어있다. 따라서 특정 데이터에 대한 추적성(traceability)을 추가적으로 제공한다.

나. 스마트 계약

1997 년 Nick Szabo 에 의해 처음 등장한 스마트 계약(Smart Contracts)은 2013 년 Vitalik Buterin 이 블록체인과 결합한 확장된 개념으로 소개함으로써 널리 알려지게 되었다. 스마트 계약은 기타 거래 내역들과 마찬가지로 블록체인 내에 트랜잭션 형태로 포함이 됨으로써, 조건이 충족되면 누구도 부인할 수 없는 자동화 된 작업을 수행할 수 있다[12].

또한 스마트 계약은 블록체인 내에 코드가 포함이되어 조건을 충족하면 해당 코드를 실행하여 블록체인 현재 상태를 나타내고 변경하는 역할을 한다. 실제 데이터는 블록체인에 전부 등록이되어 바뀔 수 없지만, 그에대한 상태를 변경할 수 있다. 이더리움 블록체인의 등장이후 스마트 계약을 사용하는 블록체인 플랫폼이 증가하였다. 이더리움은 머클 패트리샤 트리(Merkle Patrcia Tree)를 이용하여 상태를 나타내며[13], 하이퍼레저 패브릭은 월드 스테이트 데이터베이스(World State Database)를 블록체인과 함께 두어 자산의 상태 정보를 변경할 수 있다.

3. 분산 CP-ABE

속성 기반 암호화(Attribute based Encryption)는 데이터 기밀성과 유연한 접근 제어를 동시에 제공하며 클라우드 스토리지 서버에서 일 대 다 데이터 공유를 위한 기술로 사용되었다. 보통 속성 권한(AA, Atrribute Authority)와 사용자 및 데이터 소유자 그리고 클라우드 서버로 구성된다. 그 중 암호문 정책 속성기반 암호화(Ciphertext-Policy ABE, CP-ABE)는 암호문에 접근 제어 정책을 포함하며 사용자 속성 개인키를 이용한 암호화 및 복호화 기법이다[14]. AA 는 사용자의 속성을 이용하여 개인키를 발급한다. 데이터 소유자는 접근 구조를 사용하여 파일을 암호화한 다음 클라우드 서버로 업로드 한다. 사용자가 암호화된 파일에 접근하고자 하는 경우, 먼저 클라우드 서버에서 암호화된 파일을 다운로드한 다음 개인 키와 관련된 속성 집합이 암호화된 파일에 포함된 접근 정책과 일치하는지 여부를 비교한다.

초기 클라우드에서 사용된 CP-ABE 는 단일 권한(Single –Authority)이 모든 속성 개인 키 부여 권한을 가지고 있었다. 그러나 대규모 시스템에서 단일 속성 권한이 절대적으로 많은 규모의 신원을 관리하는 것은 굉장히 어렵다. 그리고 사용자로서는 키 생성에 관해 해당 단일 속성 권한에 완전히 의존할 수 밖에 없다. 만약 권한이 위협받는 경우에는 전체 시스템이 위험할 수 있다. 따라서 여러 속성

권한을 분산시켜 신원 관리의 효율성을 높이고 단일 실패 지점 위협을 없애기 위하여 [15]와 같이 분산된 CP-ABE 기법이 제안되었다.

분산된 CP-ABE 기법 수행 프로세스는 다음과 같다.

i. $\boldsymbol{globalsetup}(1^{\lambda}) \rightarrow \boldsymbol{PP} = (\boldsymbol{g}, \boldsymbol{y}, \boldsymbol{e}, \boldsymbol{p}, \boldsymbol{G}, \boldsymbol{G_T})$ 비밀 매개변수 1^{λ} 을 입력 값으로 하여 공개 파라미터 $\boldsymbol{PP}$값을 먼저 계산한다. g, y는 순환군 G의 독립된 두 개의 generator 값이다. G, G_T는 같은 위수 p 를 가지는 두개의 곱셈 순환군이며, e는 $G \times G = G_T$ 를 계산하는 겹선형 사상이다. 이 때 각각의 AA 의 집합을 $\{A_1, A_2, \ldots A_{n_a}\}$로 표현할 수 있다. 그리고 임의의 A_i가 관리하는 속성의 집합을 $\widetilde{A_i} = \{att_{i_1}, att_{i_2}, att_{i_{N_i}}\}$라고 한다.

ii. $\boldsymbol{AuthoritySetup}(\widetilde{\boldsymbol{A_i}}, \{\boldsymbol{S_{i,j}}\}) \rightarrow \boldsymbol{MSK_i}, \boldsymbol{PK_i}$ 각 AA 가 수행하며, MSK_i, PK_i를 생성한다. 결과 값은 다음과 같다.

$$MSK_i = \left\{\alpha_i, \beta_i, \gamma_i, z_{i_{j,k}} \;\middle|\; 1 \le j \le N_i, 1 \le k \le n_{i_j}\right\}$$
$$PK_i = \left\{A_i, B_i, Q_i, Z_{i_{j,k}}, T_{i_{j,k}} \;\middle|\; 1 \le j \le N_i, 1 \le k \le n_{i_j}\right\}$$

iii. $\boldsymbol{KeyGen}(\boldsymbol{MSK}, \boldsymbol{Attrs}) \rightarrow \boldsymbol{SK}$: 이는 사용자의 속성 비밀키를 생성하는 단계로써, 입력 값은 사용자의 *GID* 값인 u 와 사용자의 속성 리스트 *Attrs*, 즉 $\widetilde{U} = \{L_1, L_2, \ldots, L_d\}$이다. 각 A_i는 랜덤하게 $t_{U,i}, d_{U,i} \in Z_p$를 선택한다. 이렇게 계산되는 속성 비밀키 SK_U^i는 다음과 같다.

$$SK_U^i = \{G_{i_{j,k}} = g^{\alpha_i} g^{z_{i_{j,k}} d_{U,i}} y^{\frac{\beta_i + u}{t_{U,i}}},$$
$$= L_{i_{j,k}} = g^{z_{i_{j,k}} d_{U,i}},$$
$$= R_i = g^{\frac{1}{t_{U,i}}},$$
$$= R'_i = g^{\frac{\beta_i}{t_{U,i}}}\}$$

iv. $\boldsymbol{Encyrpt}(\boldsymbol{M}, \mathbb{w}) \rightarrow \boldsymbol{CT}$ 데이터 소유주는 자신이 암호화하고자 하는 메시지 $M \in G_T$ 에 접근 구조 $\mathbb{w}$를 적용하여 암호화 한다.

생성되는 암호문 CT는 다음과 같다.

$$CT = \{g^s, M(\prod_i \in e(g,g)^{\alpha_i})^s, y^s, \prod_{i\epsilon I} g^{Z_{i_{j,k}}(S_i)}, \prod_{i\epsilon I} e(g^{Z_{i_{j,k}}}, g^{s_i})^s \}$$

v. $\mathbf{Decrypt}(\boldsymbol{SK_U^i}, \boldsymbol{CT}) \rightarrow \boldsymbol{K}$ CT는 접근 정책 내에 인가된 속성을 가지고 있는 주체들만이 복호화 할 수 있다. 속성에 따라 생성된 비밀키 SK를 가지고 복호화한다. 식은 다음과 같다.

$$\frac{C_2 \cdot C_{i_{j,k}}^2 \cdot e\left(\prod_{i\epsilon I} R'_{i,} C_3\right) \cdot e\left(\prod_{i\epsilon I} L_{i_{j,k,}} C_1\right) \cdot e\left(\prod_{i\epsilon I} R_{i,} C_3\right)^u}{e\left(C_{i_{j,k}}^1, C_1\right) \cdot e\left(\prod_{i\epsilon I} G_{i_{j,k,}} C_1\right)}$$

$$= \frac{M(\prod_{i\epsilon I} e(g,g)^{\alpha_i})^s \cdot \prod_{i\epsilon I} e\left(g^{Z_{i_{j,k}}}, g^{S_i}\right)^s \cdot e(\prod_{i\epsilon I} g^{Z_{i_{j,k}} d_{U,i}}, g^s)}{e\left(\prod_{i\epsilon I} g^{Z_{i_{j,k}} S_i}, g^s\right) \cdot e\left(\prod_{i\epsilon I} g^{\alpha_i} g^{Z_{i_{j,k}} d_{U,i}} y^{\frac{\beta_i + u}{t_{U,i}}}, g^s\right)} \cdot e\left(\prod_{i\epsilon I} g^{\frac{\beta_i}{t_{U,i}}}, y^s\right) \cdot e\left(\prod_{i\epsilon I} g^{\frac{1}{t_{U,i}}}, y^s\right)^u = M$$

CP-ABE 를 이용하여 저작물을 암호화하면 다음과 같은 장점이 있다. 우선 일대다 복호화를 지원하기 때문에, 한 쌍의 공개 키 – 개인 키를 생성할 필요가 없다. 또한 제안하고자 하는 모델에서의 이용 허락 동의는 시스템 사용자의 속성을 기반으로 제어할 수 있다. 이 때 가장 많이 사용할 수 있는 속성으로써 사용자가 속한 플랫폼 ID, 연령층, 성별 등이 될 수 있는데 CP-ABE 는 이러한 속성들을 이용하여 정밀한 접근 제어를 제공할 수 있다. 이를 제안 모델에 적용한 방법은 다음 3 장에서 다시 논의한다.

III. 마이데이터를 활용한 블록체인 기반 저작권 관리 시스템

본 장에서는 저작권자 중심의 새로운 저작권 데이터 관리 시스템을 마이데이터 모델과 블록체인을 사용하여 설계하고자 한다. 그리고 블록체인 기반 CP-ABE 를 이용한 저작물 공유 방법에 대해 제시한다.

그리고 설계 내용을 토대로 시스템이 사용될 수 있는 이른 바 유즈 케이스(Use Case)에 대해서 분석한다.

1. 제안 시스템

본 절에서는 논문에서 제안하고자 하는 플랫폼의 전체적인 구성요소와 저작권자 중심으로 서비스 플랫폼들에 분산되어 있는 저작물 데이터를 관리하는 방법에 대해 논한다.

가. 개요

제안 모델의 목적은 플랫폼을 통해 저작권 관리와 저작물 공유를 같이 수행할 수 있는 것으로 한다. 또한 저작권자는 1 인 창작자들로 고려하였으며 이들이 보다 쉬운 방법으로 저작물 공유에 대한 제어를 할 수 있도록 한다. 기존 시스템의 중앙 서버가 없다는 점을 제외하고는 전체적인 프로세스는 유사하다. 그러나 구성요소들의 수행 역할이 어느 정도 상이한 부분이 존재한다.

1) 참여 구성요소

- 블록체인 기반 온라인 서비스 사업자(Blockchain based OSP) **:** 저작물 데이터를 서비스하는 플랫폼 사업자 조직들로, 이들의 대표적인 서버나 피어가 블록체인을 참여한다. 마이데이터 모델에서 서비스를 제공하거나 받는 플랫폼들이 모두 해당된다 [16]. 기존 플랫폼과는 달리 유통 이력 및 저작권 이용 동의와 같은

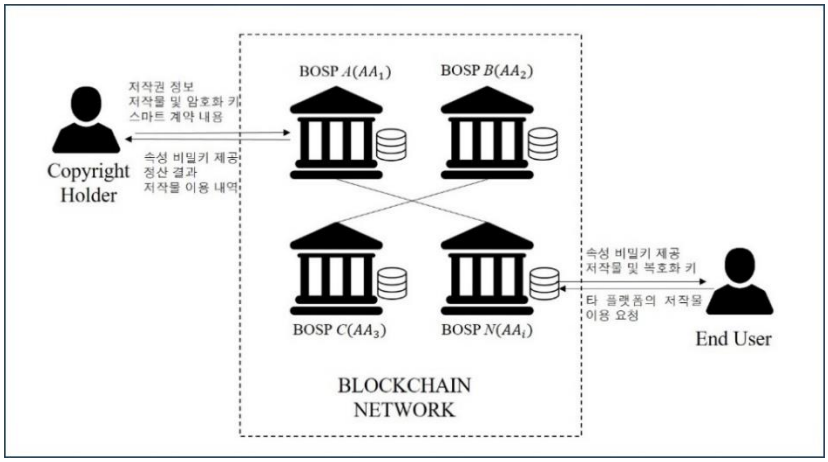
내용이 블록체인에 등록되어 네트워크에 참여하는 모든 노드들에게 공유가 되므로 위 변조와 같은 악의적인 행위를 할 수 없다.

이들은 저작권자의 데이터를 서비스할 때 반드시 저작권자의 동의(consent)가 이루어진 이후에 서로 데이터를 사용할 수 있다. 또한 이들은 저작권 이용 허락 여부를 판단하기 위한 사용자들의 속성(attribute)를 관리하고 저작물을 암호화 또는 복호화 할 수 있는 속성 비밀 키를 발급할 수 있다.

- 저작권자(Copyright Holder) : 저작권자는 권리를 보호 받아야하는 데이터 주체이다. 저작권자들은 스마트 계약을 통해 자신들의 이용허락 조건, 즉 이용허락 동의를 블록체인에 등록함으로써 저작물과 저작권 데이터 및 데이터 이용 동의 허락 정보 및 동의 상태를 게시한다. 데이터를 직접 소유하고 있지 않아도 데이터에 대한 소유권과 정산 관련 권리를 주장 할 수 있다.

- 소비자 (Data Consumer, End User) : 플랫폼이 제공하는 서비스를 통해 저작물을 공유하고 소비하고자 하는 여러 플랫폼에 분산된 다양한 사용자들이다. 이들은 거래하고자 하는 데이터를 플랫폼에서 검색한 다음, 해당 데이터를 서비스 해주는 플랫폼의 피어에게 앞에서 논한 스마트 계약을 통하여 공유 요청을 한다. 해당 플랫폼에서 소비자가 저작물 이용 기록은 블록체인에 저장된다.

다음 [그림 3]은 제안 플랫폼을 사용하는 데 있어 수행되는 프로세스를 전체적으로 나타낸 것이다.



[그림 3] 블록체인 기반 저작권 데이터 관리 시스템 구성도

i. 네트워크를 구성하는 피어는 BOSP 이며, 이들은 서비스 하는 플랫폼 사용자들에게는 AA(Attribute Authority)로서의 역할을 수행한다. 해당 플랫폼에 속한 클라이언트들의 GID(Global Id)를 바탕으로 속성 비밀 키를 생성하여 사용자들에게 부여한다.

ii. 저작권자는 저작권 정보, 이용허락 정보, 즉 저작물을 사용하기 위한 사용자의 속성 정보를 플랫폼의 프론트앤드(front-end)에 작성한다. 이 값은 트랜잭션으로 배포한다.

iii. 또한 저작권자는 AES 와 같은 대칭 키 암호화 알고리즘을 통하여 암호화 된 저작물을 자신이 속한 BOSP 의 데이터베이스에 저장한다.

iv. BOSP 는 사전에 정의된 저작물 이용허락 범위 내에서, 이용허락 동의가 된 다른 BOSP 의 플랫폼으로의 데이터 공유가 가능하다.

v. 저작물을 사용하는 데 있어서 자신이 가지고 있는 속성 비밀키로 저작물을 복호화할 수 있는 사용자, 즉 접근 권한이 있는 사용자들은 이용허락 범위 내에서 2 차 가공 또는 재 배포할 수 있다.

vi. 하나의 저작물이 공유되면, 해당 저작물에 대한 권리를 갖고 있는 저작권자 계정에 포함된다. 이로써 한 계정 정보에는 조직이 관리하는 블록체인 네트워크에 분산된 본인의 창작물 데이터 이용 기록이 저장될 수 있다.

2) 블록체인 네트워크의 형태

본 모델은 저작물의 유통 플랫폼을 호스팅하는 온라인 서비스 사업자(OSP)가. 블록체인의 피어(peer)가 되어 네트워크를 유지하고 관리한다. 사용자들은 플랫폼을 통해 조직의 피어에 접속함으로써 서비스를 이용할 수 있다.

네트워크에 클라이언트들로부터 생성된 트랜잭션이 발생하면 피어(Peer)들이 그에 대해 검증을 한다. 올바른 트랜잭션을 합의를 통해 채택하고 나면 블록에 포함되며 그 결과는 네트워크 내에 있는 노드 모두가 열람할 수 있다. OSP 는 이미 상위 인증기관에 의해 인증된 참여자들이며, 이들의 신원은 이미 네트워크에 알려져 있으므로 신뢰관계 확보를 위한 채굴, 즉 작업 증명(PoW, Proof of Work)에서 사용하는 컴퓨팅 파워를 필요로 하지 않는다. 그리고 저작권 정보는 등록 시간 즉 타임스탬프 값(Timestamp)을 중요시한다. 또한 저작권 정보가 블록체인에 한번 잘못 등록되면 수정이 어렵기 때문에 브로드캐스트 전 시뮬레이션 과정이 필요하다. 브로드 캐스트 후 분기가 발생할 것에 대해서도 고려를 해야한다. 따라서 트랜잭션을 블록에 등록하기 전, 검증하는 과정이 필요하다. 따라서 최종성(finality)을 중요시하는 비잔틴 장애 허용 (Practical Byzantine Fault Tolerance, PBFT)과 같은 알고리즘을 이용하여 합의를 수행하는 것이 적합하다. PBFT 는 네트워크 노드 1/3 이하의 노드가 장애를 일으키더라도 전체 시스템의 합의를 이끌어낼 수 있는 알고리즘이다[17].

따라서 [18]의 분류에 따라 본 모델을 구성하는 블록체인의 형태는 컨소시엄-허가형(Consoritum-Permissioned) 블록체인이다

나. 스마트 계약 구성

저작권 정보의 공유와 실제 저작물 데이터 거래 및 정산은 스마트 계약 내의 함수를 가지고 수행한다. 여기서 스마트 계약 개발자는 각 조직에서 선정된

노드들이 개발하고 검증한 다음, 블록체인 네트워크에 배포한다고 가정한다. 이들에게는 배포가 끝난 뒤에 스마트 계약의 안전성을 위해 지속적인 관리를 할 책임이 주어진다.

이를 통해 저작권 데이터를 등록하고 상태를 변경할 수 있다. 또한 데이터 이용 동의 작업이 계약 코드 수행을 통해 이루어진다. 스마트 계약은 데이터 필드(field)와 함수로 이루어져 있다.

1) 참여 구성요소 별 수행 기능

저작권자와 사용자, 서비스 플랫폼 사업자들은 스마트 계약을 이용하여 자신이 작업하고자 하는 기능을 수행할 수 있다. 다음의 [표 1]는 참여 구성요소 별로 수행할 수 있는 스마트 계약 기능을 분류한 것이다.

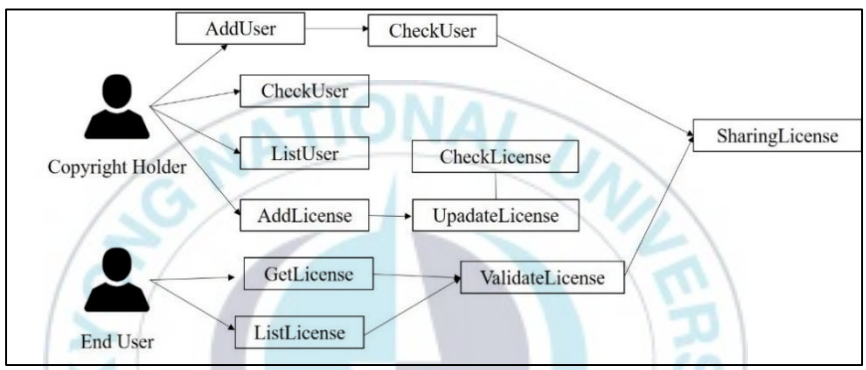
[표 1] 플랫폼 참여자 별 수행 가능한 스마트 계약 기능

구성요소	수행가능한 스마트 계약 기능
저작권자	• 저작물 정보 등록 • 저작물 정보에 대한 이용허락 정보 수정
소비자	• 사용하고자 하는 저작물 데이터 요청 • 이용허락 동의 요청 및 데이터 이용 기록 생성 • (저작권 이용허락 조건이 충족될 때) 자신이 속한 서비스 플랫폼에 원본 데이터 또는 재 가공된 데이터 재 배포
BOSP	• 배포된 스마트 계약 관리 • 플랫폼 사용자 ID 정보 업데이트 및 관리 • 사용자 ID 정보 업데이트와 동시에 플랫폼 내에 등록된 데이터에 대한 이용 동의 관리

스마트 계약은 플랫폼을 구성하는 참여자 모두가 사용할 수 있으며, 각 참여자의 플랫폼 사용목적에 따라 수행 가능한 스마트 계약 기능이 상이하다. 저작권자와 소비자는 플랫폼의 이용자로서 스마트 계약에 정의된 기능을 사용할 수 있다. 저작권자는 저작권 등록 및 저작물 이용 조건 정의와 같은 기능을 수행할 수

있으며 소비자는 저작권 및 저작물 데이터 검색 및 사용 요청과 같은 기능을 수행한다.

반면 BOSP 는 스마트 계약 코드가 포함되어 있는 블록체인을 관리하는 피어들이다. 이들은 저작권자와 소비자가 사용할 수 있도록 스마트 계약을 설계하고 생성하여 블록체인에 등록한다. 다시 말해 저작권자와 소비자 모두 BOSP 의 각 피어에 의해 합의된 스마트 계약을 참조하는 DApp 을 이용하여 원하는 작업을 할 수 있다. 저작권자와 소비자는 [그림 4]와 같이 계약의 기능을 사용할 수 있다.



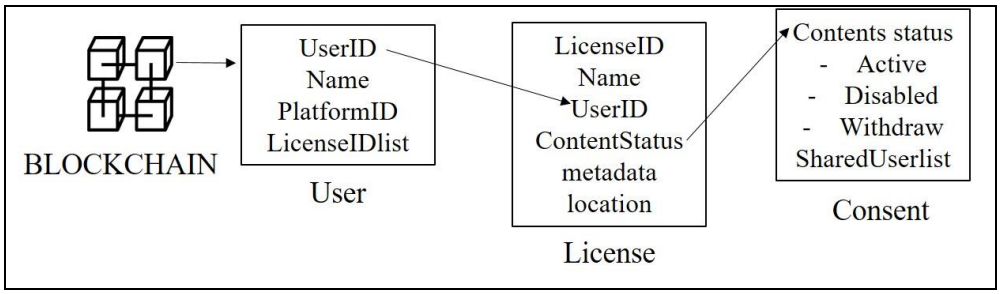
[그림 4] 스마트 계약 수행 주체 및 함수 관계

처음 저작권자 또는 소비자로부터 호출 받은 스마트 계약이 어떠한 함수를 거쳐 최종적으로 저작권 정보 및 저작물을 공유할 수 있는지가 나와있다. 각 함수의 이름은 그 기능을 나타내는 가장 대표적인 것으로써 설정할 수 있다. *CheckLicense()* 및 *CheckUser()*과 같은 함수는 프론트엔드로부터 전달받은 값이 적절한 값인지 검증을 하는 함수로써, 잘못된 저작권 정보 및 이용 로그가 기록이 되는 것을 방지한다.

2) 데이터 구조

다음은 스마트 계약의 함수와 함께 구성된 데이터 필드에 대한 것이다. 데이터 필드도 스마트 계약의 함수와 마찬가지로 각 필드마다 관계를 형성하고 있다. 각 데이터 필드 중에서도 사용자의 계정 정보 데이터 필드가 핵심으로써 이용된다. 이유는 계정 정보가 마이데이터를 기반으로 하고 있기 때문에 개인의 저작물 정보

및 이용 로그 기록 요청에 대해서 제일 처음 해당 데이터 필드로부터 응답을 받는다. 다음 [그림 5]는 각 데이터 필드 간의 관계를 나타낸다.



[그림 5] 스마트 계약의 데이터 구조체 및 참조 관계

● 사용자 계정 정보 : 블록체인 네트워크에서 사용자의 모든 신원은 마이데이터 계정을 기반으로 식별한다. 이 계정은 처음 사용자가 플랫폼에 가입할 때, 계정 ID 를 부여 받는데, 이는 고유한 값이다. 제안 모델에서 사용되는 CP-ABE 에서 데이터를 암호화 하는데 필요한 속성값을 계산하는데 신원 정보(Global ID)로써 활용된다.

모든 플랫폼의 사용자 마이데이터 계정 정보가 그림과 같은 구조체를 가지고 있다. 계정 ID 를 포함하여 계정이 속한 플랫폼의 고유 ID 와 해당 사용자가 저작권자로서 가지고 있는 저작물 리스트를 포함한다.

● 저작물 정보: 저작권자는 자신의 이름으로 여러 개의 저작물을 생성하고 배포할 수 있다. 저작물은 배포되기 전에 부여 받은 고유한 식별 번호와 실제 저작물의 위치 참조 값, 해당 데이터의 메타데이터 값이 포함되어있다. 메타데이터 값을 포함시키는 이유는 사용자로 하여금 복호화한 데이터와 블록체인 상에 게시된 데이터가 일치하는지 확인하기 위한 작업에 사용된다. 이용허락 받은 서비스 플랫폼 참여자 또는 최종 사용자는 스마트 계약을 이용하여 해당 값을 요청할 수 있다.

● 저작권자 동의 허락 정보: 저작권자는 자신이 가지고 있는 저작물 정보 내에 이용허락 정보를 매핑할 수 있다. 동의 버전(Version)과 동의 기록(Consent Record ID)및 원래의 저작권자가 배포한 저작물 ID 가 포함된다.

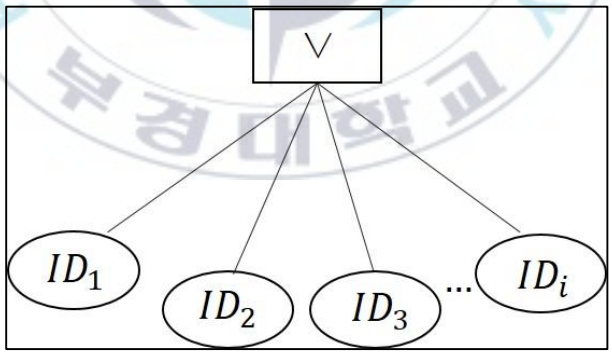
다. 블록체인 기반의 다중 권한 암호문 정책 속성 기반 암호화(Blockchain based Multi-Authority CP-ABE)

실제 저작물을 저장할 때에는, 저작물 데이터에 대한 BOSP 의 내부 변조를 방지하기 위하여 암호화를 먼저 수행한다. 이를 위해 블록체인을 기반으로 하는 분산 CP-ABE 기법을 사용할 수 있다.

실제 데이터는 AES 와 같은 대칭 키 암호화 기법의 비밀 키로 암호화 되어있고 해당 비밀 키, 즉 데이터 복호화 키를 CP-ABE 를 이용하여 암호화 한다. 이렇게 암호화된 데이터 복호화 키는 접근 정책을 만족하는 속성 집합을 가진 사용자만이 복호화 할 수 있다.

데이터 소유자, 본 모델에서 저작권자는 사용자의 속성에 기반하여 각 저작물 데이터에 대해 접근 정책을 정의할 수 있다.

본 논문에서 사용하는 CP-ABE 기법은 AND 와 OR 게이트를 지원하는 계층적인 트리 형태의 접근 구조를 적용할 수 있으며, 본 논문에서는 제안 모델의 타당성을 보여주기 위한 프로토타입의 구현을 단순화 하기 위해 가장 간단한 형태의 접근 구조를 적용하여 보여준다. 접근 구조는 다음과 같다.



[그림 6] 제안 모델에 사용된 CP-ABE 기법의 접근 정책 구조

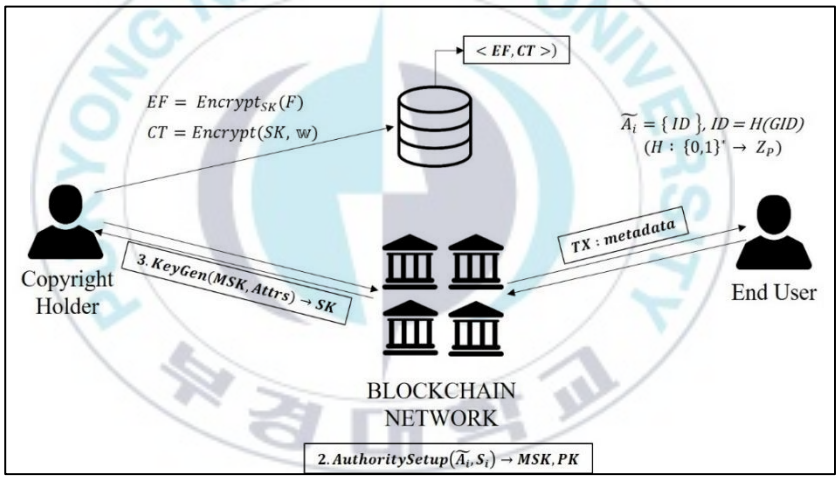
접근 구조를 위해 사용되는 속성은 사용자의 신원정보인 마이데이터 계정인 Account ID 값을 이용한다. AA 는 해당 마이데이터 계정 ID 값을 $string$값으로 입력

받아, Z_P의 원소로 랜덤하게 매핑하는 해시 함수를 이용하여 속성 값을 만들어낸다. 사용되는 충돌 저항 해시 함수 H는 다음과 같이 나타낼 수 있다.

$$\mathbf{H}: \{\mathbf{0},\mathbf{1}\}^* \rightarrow \boldsymbol{Z_P}$$

따라서 속성 값은 사용자의 마이데이터 계정 값 *GID*를 해시 함수로 처리한 $ID = H(GID)$를 이용한다.

일반적인 CP-ABE 와는 다르게 블록체인을 기반으로 하는 다중 authority 가 참여한다. 별도의 Central Authority 를 두지 않고, 바로 블록체인 네트워크를 구성하는 각 플랫폼들이 AAs(Attribute Authorities)역할을 수행한다. 다시 말해 전체 네트워크를 구성하는 플랫폼의 개수가 i개일때, i 개의 AAs 가 존재하는 것이다. 블록체인을 기반으로 한 CP-ABE 수행 단계는 다음과 같다.



[그림 7] 블록체인 기반 분산 CP-ABE

i. $\boldsymbol{globalsetup}(1^{\lambda}) \rightarrow \boldsymbol{PP} = (\boldsymbol{g},\boldsymbol{y},\boldsymbol{e},\boldsymbol{p},\boldsymbol{G},\boldsymbol{G_T})$ **:** 비밀 매개변수 1^{λ} 을 입력 값으로 하여 공개 파라미터 값을 먼저 계산한다. g, y는 순환군 G의 독립된 두개의 generator 값이다. G, G_T는 같은 위수 p 를 가지는 두개의 곱셈 순환군이며, e는 $G \times G = G_T$ 를 계산하는 겹선형 사상이다.

ii. $\boldsymbol{AuthoritySetup}(\widetilde{\boldsymbol{A_i}},\boldsymbol{S_i}) \rightarrow \boldsymbol{MSK},\boldsymbol{PK}$ **:** $\widetilde{A_i}$는 i번째 AA 가 관리하는 속성 집합이다. 접근 구조에 필요한 속성을 ID 한가지로 정의했기 때문에 $\widetilde{A_i} = \{ ID \}$이다.

그리고 $\boldsymbol{S_i}$ 는 속성에 대한 값이며, 즉 각 플랫폼에 소속되어있는 사용자의 ID 가 해당된다. 이를 이용하여 MSK, PK를 생성한다.

iii. $\boldsymbol{KeyGen(MSK, Attrs) \rightarrow SK}$: 저작권자와 소비자는 자신이 소속된 플랫폼 사업자 즉 A_i에게 속성 비밀 키의 생성을 요청한다. 이 비밀키는 블록체인의 스마트 계약으로 생성하기에는 무리가 있으므로 안전한 채널로 전달받을 필요가 있다.

iv. $\boldsymbol{Encyrpt(PK, M, Access\ Policy)}$ **:** 저작권자는 대칭 키 SK 를 이용하여 데이터 F를 암호화한다. 암호화 된 파일을 EF 라고 할 때 $EF = Enc_{SK}(F)$ 로 나타낼 수 있다. 그리고 데이터를 암호화는 데 사용한 대칭 키 SK를 CP-ABE 로 암호화 한다. 이 때 수행되는 식은 $CT = Encrypt(SK, w)$ 이다. 그 후 암호화된 데이터 EF 는 CT 값과 함께 외부 저장소에 저장한다. 후에 데이터 소비자들의 데이터 검증을 위하여 데이터의 메타데이터를 저장한다. 이미지 저작물의 경우 SHA-256 과 같은 암호학적 해시함수로 처리된 해시 값이 될 수 있다. 추가적으로 데이터 스토리지 위치 $location$값을 포함한다.

v. $\boldsymbol{\mathbf{Decrypt}(SK, CT) \rightarrow K}$ CT는 접근 정책 내에 인가된 속성을 가지고 있는 주체들만이 속성 비밀키를 이용하여 복호화 할 수 있다.

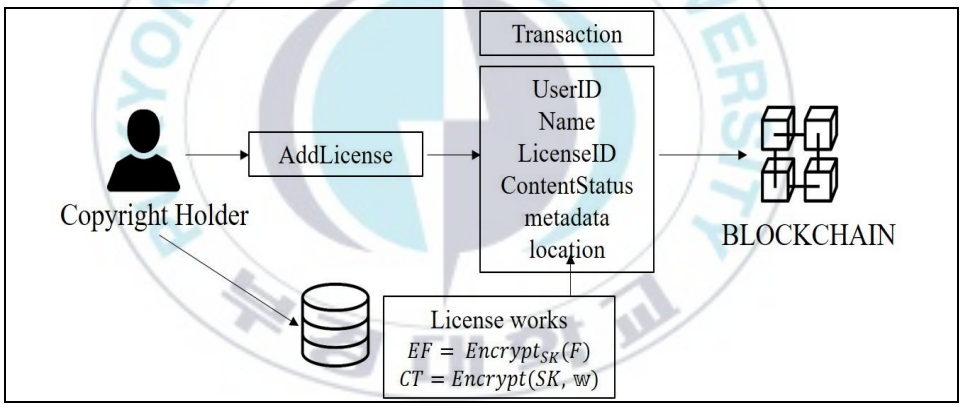
vi. $\boldsymbol{\mathbf{Dec}(SK, EF) \rightarrow F}$ 해당 속성을 가진 사용자가 키를 복호화 하고 나면 데이터를 복호화할 수 있는 대칭 키 SK를 얻을 수 있다. SK를 이용하여 EF 를 복호화하면 원본 파일 F를 얻게 되는데, 이 F가 올바른 데이터인지에 대한 검증은 블록체인에 등록된 메타데이터를 검증함으로써 확인 할 수 있다.

2. 제안 시스템의 유즈 케이스

본 절에서는 앞 서 설계된 모델을 바탕으로 플랫폼 사용자가 서비스를 이용하는 유즈 케이스 3 가지에 대해 논한다. 이 3 가지는 각각 저작권자의 저작물 등록 및 이용허락 조건 설정, 그리고 등록된 저작물에 대한 재배포, 이용동의 활성화, 마지막으로 실제 정산에 대한 근거 기록이다. 사용하는 함수와 데이터 필드는 앞서 논한 것을 바탕으로 한다.

가. 저작물 등록 및 이용허락 조건 설정

저작권자가 사용할 수 있는 함수 중 AddLicense()를 이용한 것이다. 초기의 저작권 정보를 등록할 수 있으며 그렇게 저장된 값은 제일 처음 트랜잭션을 생성하여 배포한 저작권자만이 접근을 할 수 있도록 한다.



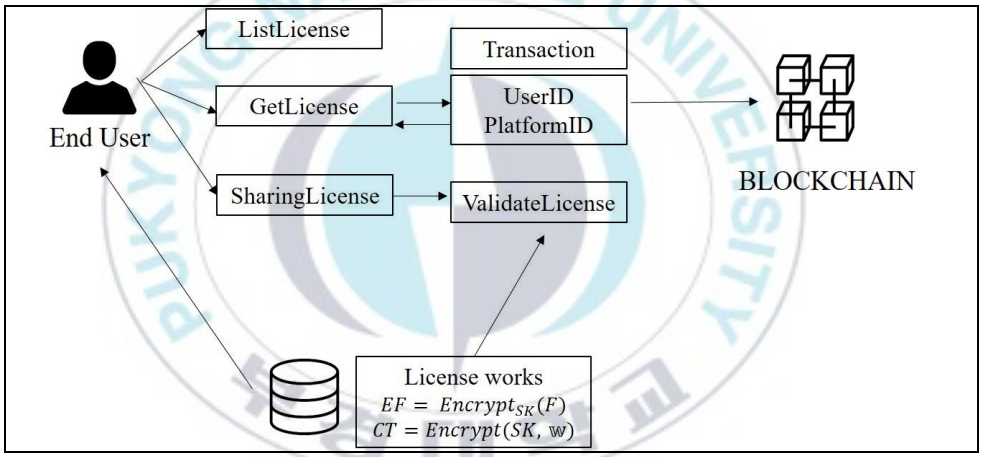
[그림 8] 스마트 계약을 이용한 저작물 등록 및 이용허락 조건 설정

저작권자 A 는 제일 처음 BOSP 의 저장소와 같은 외부 저장소에 서비스하고자 하는 저작물 *F* 를 대칭키 *SK* 로 암호화 한 후 저장한다. 그리고 스마트 계약을 이용하여 저작물 이용 허락 상태 *Consent Status* 를 정의한다. 상태 값은 Active, Withdraw, Disable 세 가지 상태로 정의할 수 있다. 또한 저작물을 암호화할 때 사용하였던 *SK* 를 CP-ABE 로 암호화한 *CT* 를 저작물과 함께 저장할 수 있다. 이 때 사용하는 접근 제어 정책은 저작권자가 서비스하고자 하는 사용자들의 속성들을

조합한 접근 구조를 사용한다. 그 후 플랫폼에 설치된 스마트 계약을 이용하여 저작권 정보를 작성하여 트랜잭션으로 제출한다. 소비자들이 저작물을 검증할 수 있게 하기 위한 메타데이터 정보 *metadata* 와 데이터 저장 위치 *location* 을 포함시킨다. 제출된 트랜잭션은 피어들의 합의 과정을 거친 후 블록체인에 등록이 되며 그 순간부터 저작권을 주장할 수 있게 된다.

나. 등록된 저작물의 사용

다음은 같은 플랫폼 또는 타 플랫폼의 사용자가 저작권 정보를 검색하고 저작물 사용요청을 하는 경우이다. 요청과 검색은 별도의 접근 제어를 하지 않는다.



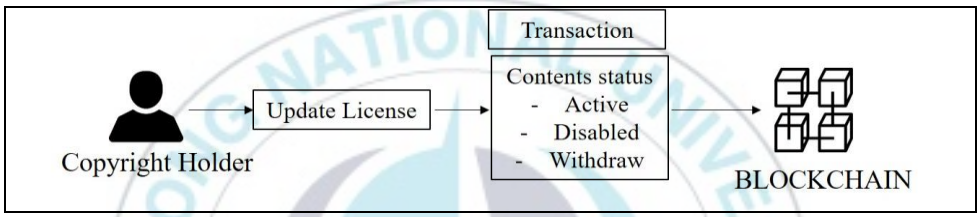
[그림 9] 등록된 저작물의 사용 및 재 배포

블록체인 네트워크 내에 또 다른 BOSP 가 호스팅 중인 플랫폼 사용자이자 소비자인 B 는 플랫폼에 자신이 사용하고자 하는 저작물을 검색한다. 그리고 A 의 저작물을 사용하고자 한다. 그 후 B 는 스마트 계약을 이용하여 사용 요청 트랜잭션을 생성하여 배포한다. 해당 트랜잭션은 A 의 저작물을 관리하고 있는 BOSP 에게도 전달이 되고 피어들은 블록체인에 등록된 A 의 이용 동의 상태를 확인할 수 있다. 저작권에 명시된 이용 동의 접근 정책에 근거하여 B 가 동의 가능한 사용자라면 A 는 접근 정책에 B 의 *ID* 를 포함시켜 재 암호화한다. 그리고 데이터

사용 정책에 대해 승인이 난다. 그 뒤 트랜잭션의 반환 값으로 저작권 정보를 얻은 B 는 CP-ABE 로 복호화하여 얻어낸 대칭 키 *SK* 로 데이터를 복호화 한 후 블록체인에 등록된 메타데이터 값과 비교하여 실제 게시된 데이터의 무결성을 검증할 수 있다. B 의 사용 동의 후 저작권자 A 의 계정 정보에는 B 의 이용 기록이 포함되어 있으며 이를 통해 이용 위반 추적이 가능하다.

다. 이용허락 변경 및 철회

저작권자는 자신의 데이터 사용 허락이 정의된 계약의 상태를 트랜잭션을 보내 변경할 수 있다.



[그림 10] 이용 허락 내용 변경

1) 활성화 → 비활성화(Active → Disabled)

저작권자가 자신의 의지에 따라 앞으로 해당 저작물에 대한 사용 조건을 '사용할 수 없음'으로 바꾸는 것이다. 트랜잭션 생성 후 합의에 의해 해당 트랜잭션의 검증이 완료되고 나면, 전체 장부의 상태 값(world state)값이 변경된다. 즉 네트워크 전역에 해당 저작권자가 이용허락 조건을 변경하였음을 알리게 되는 것이다. 이후에 데이터 이용허락 기존의 데이터 사용자들은 가지고 있는 데이터를 다시 다운로드 받을 필요 없다. 스마트 계약의 함수나 기능의 내용이 변경된 것이 아니라 내부에 있는 필드 값이 변경된 것이기 때문이다.

2) 활성화, 비활성화 → 철회(Active, Disabled → withdraw)

한편 저작권자가 모종의 이유로 자신의 저작물을 이용허락 조건과 상관없이 배포를 중단하는 경우도 발생할 수 있다. 마이데이터 모델 관점에서 살펴보면 이는

이용 허락 조건이 철회(withdraw)상태로 전환되는 것이다. 철회를 하기 위해 실제 스마트 계약 코드를 블록체인에서 완전히 제거하는 것은 불가능하지만 계약내 조건을 '더 이상 유효하지 않은 상태'로 변경하면 된다[19].

실제로 이용허락 철회 후 사용자들이 이미 다운로드 받은 데이터를 철회 되고 나서 완전히 오픈 데이터(Open Data) 화 하거나 또는 완전히 배포금지 하는 두 가지 경우에 대해 고려해 볼 수 있다. 전자의 경우 철회는 더 이상 원 저작권자에게 소유권 및 정산 권리가 없음을 의미한다.

라. 실제 데이터 거래 및 정산

블록체인을 적용한 저작권 관리 모델이 갖는 가장 큰 장점은 사용자들의 저작물 이용기록과 그에 대한 수익이 네트워크에 참여하는 모든 노드가 열람할 수 있다는 것이다. 플랫폼의 스마트 계약에는 저작권자와 플랫폼 사이의 수익률 배분 규칙이 미리 설정되어 있으며 이를 블록체인에서 확인할 수 있다. 저작권자의 계정에는 수익을 계산하여 저장하는 자산을 포함시킨다. 그리고 저작물의 거래가 발생할 때 마다 각 플랫폼에서 발생된 수익을 계약의 함수가 자동적으로 계산하여 값을 변경할 수 있다. 이로써 저작권자는 네트워크를 이루는 플랫폼들에 공유된 자신의 저작물로 발생한 모든 수익을 관리할 수 있다.

실제 정산 모델은 채택하는 블록체인의 형태에 따라 조금씩 상이할 수 있다. 암호 화폐를 유통하는 블록체인의 경우, 해당 계정에 화폐를 입금하는 등의 프로세스 수행을 통하여 실제 정산까지 가능하다. 본 모델에서 사용하는 블록체인의 경우 암호화폐를 필요로 하지 않는 모델을 사용한다. 이는 현실에서의 정산에 대하여 블록체인에 포함된 정산 기록을 플랫폼이나 저작권자 그 누구에게도 조작되지 않은 무결성이 보장된 근거로 사용할 수 있다.

IV. 저작권 데이터 공유 플랫폼 스마트 계약 구현 및 분석

본 장에서는 앞 장에서 구체적으로 논의된 모델을 실제로 블록체인 네트워크를 구축하고, 스마트 계약을 설계함으로써 구현 내용에 대한 동작 확인 후, 기존의 저작권 관리 시스템과 비교 분석한다.

먼저 제안 모델 기반이 되는 컨소시엄-허가형 블록체인의 설계를 위하여 조직 중심의 네트워크를 구축한 다음, 각 조직의 피어들이 블록의 합의 결과를 도출하는 방식을 채택한다. 하이퍼레저 패브릭(Hypereledger Fabric)은 기업형 블록체인을 개발하는데 현재 알려진 플랫폼 중 가장 적합하며 제안 모델 플랫폼 구현 방식으로 채택하였다.

1. 구현 내용

가. 개발 환경

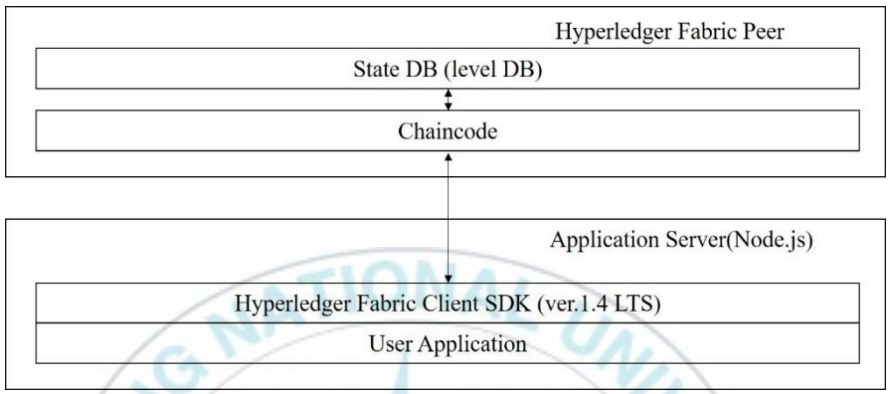
- CPU : 6 processors cores of CPU on Oracle VM Virtualbox 6.0.12
- 운영체제 : Ubuntu 18.0.4.LTS
- 메모리 : 4GB
- 하드디스크 : 20GB
- 개발언어 Go Language
- 오픈 소스

  - Package : github.com/hyperledger/fabric/core/chaincode/shim[20]
  - Source Code : DRM.go[21]
- 참고자료 : Hyperledger Docs[22], IBP(IBM Blockchain Platform)[23]

나. 모델 설계

1) 시스템 구조

다음의 [그림 11]은 전체적인 응용프로그램의 구조를 나타내는 것이다. 크게 Application Server 와 Hyperledger Fabric Peer 로 나뉜다.



[그림 11] 응용프로그램 구조

Node.js 를 이용하여 서버에 웹 UI 를 구현하여 서버 측에서 하이퍼레저 패브릭 클라이언트 SDK 를 통하여 하이퍼레저 패브릭에 접근할 수 있다. 체인코드는 하이퍼레저 패브릭에 접근할 때 사용되는 방법을 구현한 것이며 이를 이용하여 상태 DB 의 데이터 상태 조작이 가능하다.

본 모델에서 사용하는 상태 DB 는 해시맵(HashMap)과 같은 역할을 수행하는 데이터베이스이다. 데이터는 [그림 12]와 같이 2 차원적으로 저장된다[24].

```
{“License ID”{“LicenseID ”:“0000012x”.“LicenseName”:“image_01”, “userID” :
“_CopyrightHolderID”, “TimeStamp:” : “2019-11-04T05:25:56.1624733363+09:00”,
“metadata” : “xxxxxx”}}
```

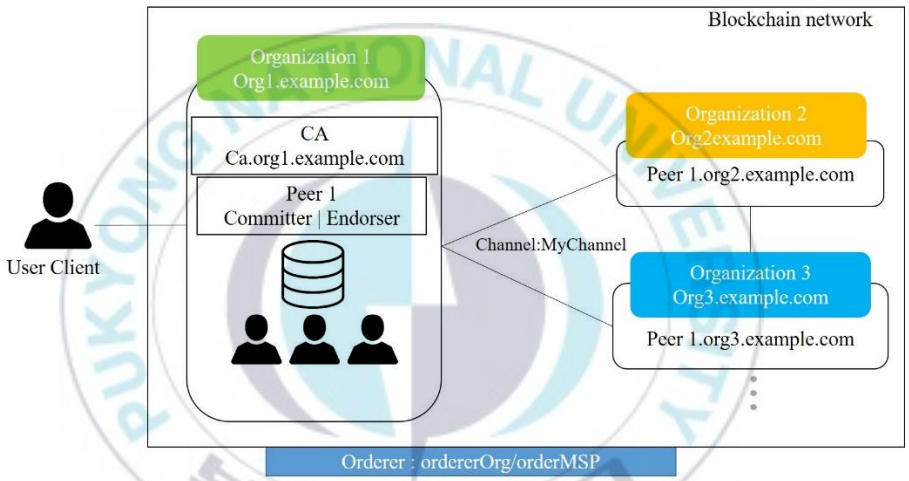
[그림 12] 데이터 구조

작성된 체인코드를 테스트하기 위해서는 테스트 데이터가 필요하다. 또한 하이퍼레저 패브릭은 트랜잭션 배포-시뮬레이션- 과 반환-실제 배포-등록을 거쳐 블록이 완성되어 장부에 등록이 되기 때문에, 블록의 최종성을 보장한다. 작은 수의 노드가 참여하여 다른 블록체인 플랫폼보다는 높은 TPS 를 특징으로 하지만, 오더러와 커미터를 거쳐 합의가 수행되기 때문에, 체인코드 실행 후 데이터 반영이

즉각적으로 이루어졌을 가능성은 낮다[25]. 따라서 체인코드 실행 후 state DB 에 저장된 데이터 상태가 변경되고, 블록체인에 기록이 되면 피어로부터 SDK 를 통해 다시 사용자 어플리케이션에 해당 결과를 반환되는 이벤트를 구축해야 한다.

2) 패브릭 네트워크 분석

하이퍼레저 패브릭 블록체인 네트워크를 구성하기 위해서 참여 조직부터 구성할 수 있다. 앞 서 논한 것처럼 각 대표 피어가 있고, 정렬자 노드(Orderer), CA 가 존재한다. 그 내용은 다음 [그림 13]과 같다.



[그림 13] 전체 네트워크 와 조직 1

Organization1 은 서비스 플랫폼 사업자들이다. 조직에게는 각 조직을 대표하는 피어(peer)들이 존재한다. 플랫폼은 저작권자와 소비자가 포함된 사용자들이 가입해 있다. 사용자들은 클라이언트서 트랜잭션을 발생시키며, 피어들은 커미터(Committer)나 검증자(Endorser)가 되어 해당 트랜잭션들을 먼저 검증한 후 합의를 도출한다. 정렬자 노드는 합의 당시 순서대로 트랜잭션을 정렬하는 작업을 수행한다. 피어 및 정렬자는 단일 개체가 아니라 블록체인 네트워크에 분산 시킨 형태로 사용할 수 있다.

3) 스마트 계약 분석

하이퍼레저 패브릭 플랫폼에서 사용하는 스마트 계약 코드는 Go 언어나 Python 으로 작성 될 수 있다. IBM 은 체인코드 개발을 위한 자신들만의 Blockchain Package 를 다양한 IDE 를 통해 배포하고 있으며, 이를 이용하여 플랫폼 관리자들 또는 개발자들은 간편하게 개발할 수 있다.

체인코드는 피어 컨테이너와는 별도의 컨테이너로 작동하며, 그 안에 특정언어로 작성된 스마트 계약 코드가 포함되어 있다[26]. 좀 더 정확히 말하자면, 스마트 계약은 하이퍼레저 플랫폼에 사용되는 프로그램 논리(logic)를 구현한 코드이고, 계약 코드를 모아 컨테이너 단위로 관리하는 것을 체인코드라고 한다.

본 모델에서 사용하는 스마트 계약 코드는 세 가지 부분으로 나누어 각각 *main.go, chaincode.go* 로 구현되었다.

- 실행 파일 정의

main.go 는 체인코드가 실제로 실행할 파일로써, *main* 함수의 *shim.start()*를 실행함으로써, 피어와의 통신을 수행하고 스마트 계약 내 함수 실행이 이루어진다. *main.go* 의 내용은다음과 같다.

```
func main(){
        err := shim.Start(new(chaincode.CopyrightmgCC))
        if err != nil{
        fmt.printf("Error in Chaincode process : %s", err)
        }
}
```

[그림 14] main.go 의 shim.start()

● 자산(asset) 정의

Chaincode.go 에는 구조체 형태로 자산이 정의되어 있다. 자산은 구조체로 정의되어 있으며 플랫폼 사용자 *User* 와 저작권 정보 *License* 가 있다.

```
type User struct {
        userID     string
        platformID string
        Name       string
        nOfworks       int
        wallet       int
}

type License struct {
        Id              string
        Name            string
        CopyrightHolderID       string
        Consentstatus       string
        metadata            []bytes
        _url                 string
        Timestamp        time.Time
}
```

[그림 15] 사용자 및 저작권 정보 정의

.*User* 는 본인의 이름, ID 와 소속 플랫폼 ID, 이름과 블록체인에 등록된 저작물 수를 포함한다. 그리고 *License* 는 마찬가지로 저작권 ID 와 저작물의 이름, 소유주 및 이용동의 상태 *Consentstatus* 를 포함한다. *metadata* 와 *_url* 은 각각 저작물 메타데이터 및 저작물 저장 위치를 나타낸다. 마지막으로 *Timestamp* 값은 *License* 의 최초 등록시간이다.

● 저작권 등록 및 저작물 공유 함수 정의

앞서 3 장의 [그림 4]에서 논의된 스마트 계약 내의 함수 구조를 따라 *impl.go* 내에 설계한다. *impl.go* 를 실행함으로써 체인코드를 호출한 피어 내에 포함된 월드 스테이트 데이터베이스와의 직접적인 입•출력이 수행된다. 9 개의 함수 중 가장 핵심적인 기능을 하는 *AddLicense(), GetLicense(),* 그리고 *SharingLicense()*에 대해서만 논한다.

또한 해당 함수를 실행하는 주체들은 이미 인증기관으로부터 인증된 플랫폼에 가입되어 있는 사용자들이다. 이들 신원의 등록은 함수 *AddUser()*에 정의되어 있는 대로 수행한다.

*AddLicense()*는 자신의 저작물을 올리고자 하는 유저가 *LicenseKey* 를 등록할 수 있는 함수이다. *Chaincode.go* 에 정의되어 있는 *struct License* 내의 필드 값 만큼을 입력받는데, 이 때 매개변수 수가 모자라면 *err* 를 반환하고 체인코드 컨테이너를 종료한다.

정상적으로 매개변수를 받았다면, 부가적인 값에 대해서도 입력을 받는데 차례대로 *holderPercentage, platformPercentage* 이다. 이는 각각 저작권자가 가지는 수익률, 마이데이터 플랫폼이 가지는 수익률이다.

LicenseBytes 는 입력 값으로 들어온 JSON 타입의 저작물 데이터를 마샬링(marshaling)하여 저장한 값으로써, *License Key* 와 함께 *putstate()*를 이용하여 월드 스테이트에 기록한다. 모든 과정은 체인코드 트랜잭션 유효성 검증 후 블록체인에 기록 된다.

*AddLicense()*의 내용은 다음과 같다.

```
func (this *CopyrightmgCC) AddLicense(stub shim.ChaincodeStubInterface, params
[]string) sc.reponse {

        holderPercentage int
        platformPercentage int
        licenseID string
        license *License
        err error

        holderPercentage, err = strconv.Atoi(string(params[2]))
        if err != nil { return shim.Error(err.Error()) }

        platformPercentage, err = strconv.Atoi(string(params[3]))
        if err != nil { return shim.Error(err.Error()) }

        licenseBytes, err = json.Marshal(license)

        licenseID, err = this.getLicense (stub, params[0])
        if err != nil {
```

```
                return shim.Error(err.Error())
        }
        err = stub.PutState(licenseID, licenseBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Success in %s recorded\n", params[0])

        return shim.Success(nil)
}
```

[그림 16] 구현된 AddLicense()의 일부

*GetLicense()*는 소비자가 저작권 플랫폼에서 자신이 사용하고자 하는 저작권 내용에 대하여 검색하고 정보를 얻는 함수이다. 키 값이 되는 *Licnese* 를 이용하여 검색하며, *usetype* 을 입력하면, 조건에 따라 *get* 할 수 있다. *GetLicense()*의 내용은 다음과 같다[21].

```
func (this *CopyrightmgCC) GetLicense(stub shim.ChaincodeStubInterface, params
[]string) sc.reponse {

        err error
        useType string

        useType = string(params[0])
        queryString :=
        `{
                "selector": {
                                "useType": ` + useType + `
                }
        }`
        fmt.Printf("queryString:\n%s\n", queryString)

  // Invoke query
        resultsIterator, err := stub.GetQueryResult(queryString)
        if err != nil {
                return shim.Error(err.Error())
        }
        defer resultsIterator.Close()

        var buffer bytes.Buffer
        buffer.WriteString("[")
```

```
        // Iterate through all returned assets
        bArrayMemberAlreadyWritten := false
        for resultsIterator.HasNext() {
                queryResponse, err := resultsIterator.Next()
                if err != nil {
                        return shim.Error(err.Error())
                }
                if bArrayMemberAlreadyWritten == true {
                        buffer.WriteString(",")
                }
                buffer.WriteString("{\"Key\":")
                buffer.WriteString("\"")
                buffer.WriteString(queryResponse.Key)
                buffer.WriteString("\"")

                buffer.WriteString(", \"Record\":")
                buffer.WriteString(string(queryResponse.Value))
                buffer.WriteString("}")
                bArrayMemberAlreadyWritten = true
        }
        buffer.WriteString("]")

        fmt.Printf("%s ", buffer.String())

        return shim.Success([]byte(buffer.String()))
}
```

[그림 17] 구현된 GetLicense()

함수 *Sharing Lincense()*는 저작물을 공유하는 함수이다. 블록체인에 기록된 저작권자 정보 및 수익률 그리고 정산 근거, 수수료 계산 결과 및 저작물을 공유함으로써 다시 계산된 수익금등이 월드 스테이트로부터 읽어오거나 새로이 쓰여진다. *SharingLicense()*는 *CheckLicense()*함수를 동반하여, 해당 *License* 에 등록된 저작물 정보가 올바른지 검사를 수행한다. 이전 단계까지 완전히 완료되고 나면, *stub.Putstate()*를 통해 월드 스테이트에 기록한다.

*Sharing License()*의 내용은 다음과 같다.

```
func (this *CopyrightmgCC) SharingLicense(stub shim.ChaincodeStubInterface, params []string) sc.reponse {

        LicenseKey, err = this.getLicenseKey(stub, licensename)
        if err != nil {     return shim.Error(err.Error()) }

        licenseBytes, err = stub.GetState(licenseKey)
        if err != nil { return shim.Error(err.Error()) }

        err = json.Unmarshal(licenseBytes, &license)
        if err != nil {     return shim.Error(err.Error()) }
        ……
        sharingLicense = &SharingLicense{
                name,
                licenseeUserId,
        }
        sharingLicense,, err = json.Marshal(sharingLicense)

        sharingLicenseKey, err = this.get sharingLicense (stub, licensename, licenseeUserId)
        if err != nil {
                return shim.Error(err.Error())
        }
        err = stub.PutState(sharingLicense, sharingLicenseBytes)
        if err != nil {
                return shim.Error(err.Error())
        }
        fmt.Printf("Sharing License “From %s to %s is complete”,licensename, licenseeUserId)
        return shim.Success(nil)
}
```

[그림 18] Sharing License()의 일부 코드

다. 실행 결과 화면

본 절은 앞서 설계된 모델 내용을 바탕으로 스마트 계약 구현 결과에 대한 내용을 크게 네트워크 설정 상태와 체인코드 실행 결과 두 부분으로 분류하여 논하도록 한다.

1) 네트워크 설정

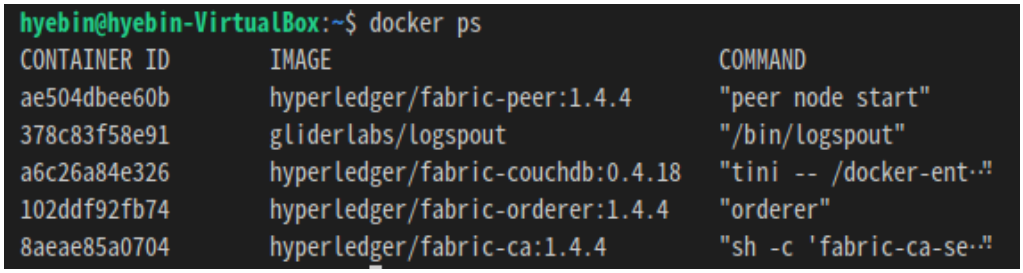
블록체인과 상태 데이터베이스에 스마트 계약을 이용하여 어떠한 기록을 하기 전, 네트워크의 설정을 모두 마쳐야 한다. 네트워크 상태는 패브릭 네트워크 설정 경로 하위에 있는 *crypto-config.yaml* 파일로 확인할 수 있다.

```
- &Org1
    # DefaultOrg defines the organization which is used in the sampleconfig
    # of the fabric.git development environment
    Name: Org1MSP

    # ID to load the MSP definition as
    ID: Org1MSP

    MSPDir: crypto-config/peerOrganizations/org1.example.com/msp

    AnchorPeers:
        # AnchorPeers defines the location of peers which can be used
        # for cross org gossip communication.  Note, this value is only
        # encoded in the genesis block in the Application section context
        - Host: peer0.org1.example.com
          Port: 7051
```

[그림 19] 네트워크 구성

네트워크는 한 개의 정렬자 노드와 한 개의 피어, 월드 스테이트인 Couch DB 및 하나의 CA 로 구성되어 있다. 조직의 MSP 는 crypto-config/peerOrganizations/org1.example.com/msp 에 저장되어 있다.

네트워크 설정을 마친 후, 계약 함수의 수행을 확인하기 위해 본 논문에서는 Visual Studio Code 의 IBM Blockchain Platform(IBP) Package 를 사용한다. 패키지를 이용하여 로컬 패브릭 네트워크를 구축하면 [그림 19]의 설정과 같이 하나의 조직 하에 하나의 피어 및 오더러 그리고 앵커 피어(Anchor Peer)가 설치된다. [그림 20]은 IBP 를 이용하여 네트워크 기동을 마친 상태이다.



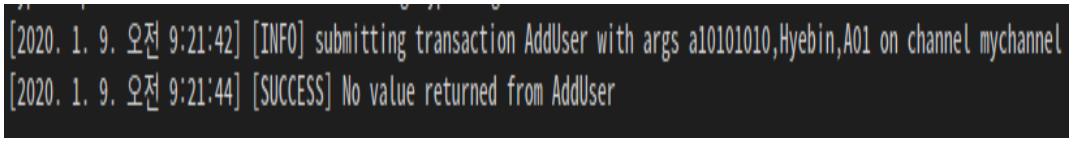
[그림 20] 기동된 패브릭 네트워크 도커 이미지

그리고 터미널에서 *docker ps* 명령을 이용하여 기동된 조직 컨테이너의 세부사항을 살펴볼 수 있다.

2) 스마트 계약 실행 결과

스마트 계약은 별도의 체인코드 컨테이너를 Org1 의 Peer1 과 연결하여 실행할 수 있다. *AddUser()*와 *AddLicense(),* 그리고 *SharingLicense()* 이렇게 총 3 개의 함수에 대한 구현과 동작 결과를 살펴보았다.

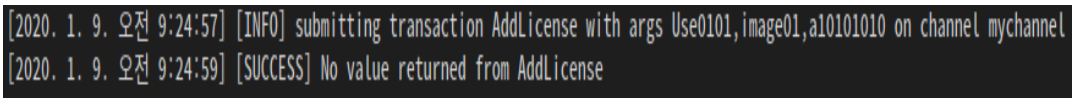
체인코드의 설치 후 피어가 참여하고 있는 채널 myChannel 에 먼저 AddUser()에는 사용자 계정(ID)를 Key 값으로 하는 Write Set 이 생성된다. 이 때 값은 {ID = “CCC”{ID = “a10101010”, Name = “Hyebin”, PlatformID = “A01”, RegData=”2020106163731”}} 이렇게 기록된다. [그림 21]은 위의 값으로 트랜잭션을 제출하고 나서 성공했다는 로그를 콘솔에서 출력한 것이다. 이 때 반환 값은 별도로 존재하지 않는다.



[그림 21] Add User 결과 화면

다음은 *Add License()*에 관한 내용이다. *AddLicense()*는 저작물 ID 를 key 값을 매개변수로 하고 있으며, 이 때 값은 {LicenseID = “Use0101”{LicenseID = “Use0101”, LicenseName = “image01”, UserID = “a10101010”, RegData=”2020106163731”}}

이렇게 기록된다. [그림 22]는 마찬가지로 위의 값으로 트랜잭션을 제출하고 나서 성공했다는 로그를 콘솔에서 출력한 것이다. *RegData* 로 *Timestamp* 값을 저장하며 이는 추후에 저작권 등록 시점의 근거로 사용할 수 있다.



[그림 2 2] AddlLicense 결과 화면

*AddUser()*와 *AddLicense()*의 결과 값은 RDBMS 처럼 관계를 갖고 있는 것이 아니라 Key 값인 *UserID* 를 공유하고 있다. *UserID* 내에 포함된 정보로 *LicenseID* 값이 있으며 해당 *LicenseID* 는 또한 이용자 정보를 포함하여 동의 관리를 할 수 있다. 따라서 데이터가 테이블 형태로 참조하고 있지 않아도 *UserID* 를 이용하여 키 쿼리가 가능하다. 이 값을 바탕으로 *ShainrgLicense()*에 명시되어 있는 코드를 이용하여 수익을 계산하고 계정 정보에 저장하며 *stub.Getstate()*를 통해 마찬가지로 저장된 값을 호출할 수 있다.

2. 분석 및 평가

가. 저작권 관리

기존의 저작권 관리 모델은 OSP 의 서버 및 데이터베이스에 저작물 이용 기록과 저작권 관련 정보를 저장해두고 이용하는 방식을 채택해왔다. 저작권자는 이들에게 위탁함으로써 자신의 저작물을 플랫폼 사용자들에게 쉽게 알림으로써 배포할 수 있었고, 사용자는 OSP 가 제공하는 플랫폼에서 원하는 저작물을 검색하고 이용할 수 있었다. 앞서 논한 것과 같이 조직 중심의 클라이언트-서버 구조는 서버가 단일 실패 지점이 될 위험성이 높다. 또한 조직 내부의 악의적인 관리자가 자신들의 이익을 위하여 데이터 이용 로그를 위 • 변조할 수 있다는 단점이 존재하였다. 이를 보완하기 위한 방안으로써 제안된 블록체인 기반 저작권 관리 모델의 특성은 다음과 같다.

1) 기존 중앙 집중형 구조 저작권 관리 모델과의 비교 분석

- 무결성(Integrity): 저작권 데이터의 등록 당시의 시간 값(Timestamp)와 저작권자 그리고 저작물 메타데이터(metadata)값이 블록체인 상에 등록되면, 이 값은 블록체인의 특성으로 인하여 변경할 수 없다. 또한 저작물 데이터는 암호화된 상태로 외부 데이터베이스에 저장이 되기 때문에, 악의적인 위•변조를 방지할 수 있다.

- 불변성(Immutability): 블록체인을 구성하는 블록과 그에 포함된 트랜잭션들은 변경할 수 없다. 따라서 한번 저작권 데이터가 등록되면, 해당 데이터의 상태(state)가 변경될 수는 있어도, 그에 대한 기록(history)가 삭제되지는 않는다. 다시 말해, 저작권의 이용 동의 상태가 변경되어도 저작권이 존재 여부에 대한 과거 기록이 변경 또는 삭제되지는 않는다.

- 투명성(Transparency): 블록체인을 구성하는 노드들, 즉 읽기 권한이 있는 노드들은 네트워크에 배포되거나 등록된 트랜잭션들을 열람할 수 있다. 이들 모두가 트랜잭션을 실시간으로 열람할 수 있게 됨으로써, 악의적인 노드들이 임의로 트랜잭션 내부 입력(input) 또는 출력(output) 값을 변경하지 못하게 된다.

- 책임 추적성(Traceability): 블록체인내에는 계정을 기반으로 자신의 모든 이용 및 데이터 요청 기록이 포함되어 있다. 이를 이용하여 이용자의 이용 위반 행위 추적이 가능하다.

위와 같이 저작권 관리 모델을 블록체인을 적용함으로써 얻는 이득은 다양하다. 특히 지금까지 연구된 블록체인 기반 저작권 및 저작물 공유시스템은 주로 중개자 없는 C2C 모델을 채택하였으며 관련 플랫폼이 많이 등장하였으나 기존 플랫폼과 호환이 되지않거나 서비스를 하는 데 어려움을 겪는 등의 문제가 다수 존재했다.

2) 기존 블록체인 기반 저작권 관리 모델과의 비교 분석

이론적으로 C2C 모델은 가장 이상적인 저작권자와 소비자 간의 저작물 공유 시스템을 이룰 수 있다. 그러나 실제 저작권 관리 시스템은 신탁관리단체와 서비스 사업자와 같은 다양한 중개자들이 참여하며 이들을 완전히 배제하는 것은 현실적으로 불가능하다. 따라서 개개인의 사용자가 하나의 노드가 되는 것이 아닌, 조직이 노드가 되어 클라이언트 역할을 하는 사용자에게 블록체인 기반 서비스를 제공하는 모델도 연구되었다. 개인의 노드가 아닌 조직 중심의 네트워크를 이루는 관계로 모델의 현실성이나 확장성은 충족되었으나, 기록의 투명성은 다시 낮아지게 되었다.

또한 조직 중심의 블록체인 모델은 저작권자 개인이 스마트 계약을 적극적으로 설계하여 배포하는 점 보다는 이미 배포된 저작권 정보에 대한 투명성 보장을 중요시하였다. 다시 말해 본래 서비스 플랫폼이 가지고 있던 데이터 이용 기록을 탈중앙화 시켜 단일 실패 지점 위협을 낮추고, 단일 플랫폼내에서의 위조를 방지하여 기록에 대한 부인 방지가 가능하다는 점을 중점으로 한다. 제안 모델은 투명성 제공을 포함하여 저작권자 개인이 조직들 간의 블록체인 네트워크에서 분산되어 있는 자신의 저작물 정보 기록을 모아 보다 적극적으로 관리할 수 있도록 마이데이터 개념을 적용하였다. 이렇게 함으로써 일부 탈 중앙화 모델임에도 불구하고, 저작권자 중심의 저작권 관리 및 저작물 공유에 대한 이용 동의 허락 계약 관리를 수행할 수 있다[27].

[표 2]는 기존의 중앙 집중형 구조의 저작권 관리 모델과 블록체인 저작권 관리 모델 그리고 제안 모델의 특징을 비교 분석한 결과를 나타낸다.

[표 2] 저작권 관리 모델 비교 및 분석

	기존 중앙집중형 저작권 관리 모델	블록체인 기반 저작권 관리모델	마이데이터를 이용한 블록체인 기반저작권 관리 모델
네트워크 유형	클라이언트-서버	블록체인 기반의 P2P 네트워크	블록체인 기반의 P2P 네트워크
위협 요소	단일 실패 지점(SPoF)	악의적인 조직내부 노드 또는 합의 노드	합의 노드
읽기 권한	△ → 서버가 제공하는 정보를 전적으로 신뢰해야 함	O	O
쓰기 권한	△ → 기록되어 있는 내용을 직접적으로 변경하지는 못함 : 서버에 요청하는 방식	△ → 직접 스마트 계약을 작성 할 수 있음(C2C) → 투명한 기록을 열람할 수 있다는 것에 중점을 둠(B2C)	O → 저작권자 본인이 이용조건을 스마트 계약을 이용하여 작성 가능
무결성 보장	X → 내부 직원에 의한 위조 가능성	O	O
사용자들의 접근 난이도	쉬움	(C2C)모델의 경우 어려움	쉬움 → 기존 중앙집중형 저작권 관리 모델을 사용할 때와 유사

블록체인 기반 저작권 관리 모델에 마이데이터를 적용하며 네트워크 피어들을 조직 플랫폼으로 설정한 것은 여러 장점이 존재한다. 다양한 종류의 수많은 플랫폼이 존재하지만 저작권자는 보통 한 플랫폼에 종속되어 창작 활동을 한다. 문제는 해당 플랫폼이외의 다른 플랫폼의 사용자가 이를 가져다 사용할 경우 그의 활동을 추적할 수 없다는 점이다. 자신도 모르는 사이에 저작권 침해가 발생할 경우가 높고 이를 방지하기 위하여 플랫폼간의 블록체인 네트워크를 구성하여

저작권 정보를 등록하고 네트워크에 참여하는 플랫폼 사용자들은 어느 플랫폼에서든 저작물 이용 동의 정보를 확인할 수 있도록 하였다. 플랫폼의 입장에서는 자신들만이 서비스하고 있는 데이터를 다른 플랫폼과 공유를 하게 되는 것이다. 하지만 점점 저작물의 형태가 단일하지 않고 여러 분야에 걸쳐 융합된 다양한 저작물이 등장하고 있으며 이를 위해 다양한 저작권자들이 서로가 소비자의 입장에서 공유하고 거래하여 창의적인 협업을 이뤄낼 수 있다. 제안 모델은 더 나아가 플랫폼들은 이들을 위해 서비스 플랫폼을 개선시켜 제공함으로써 가치를 높이고, 저작권자는 더 다양한 창작을 함과 동시에 자신이 제공하는 데이터 이용 기록을 관리할 수 있으며, 소비자들은 본인이 가진 속성에 맞는 다양한 창작물들을 접하고 공유할 수 있다.

나. 블록체인 기반 CP-ABE 를 이용한 저작물 공유

제안 모델은 블록체인 기반의 CP-ABE 를 이용하여 암호화 된 키를 저작물 복호화 할 때 사용한다. 이는 기존의 분산된 다중 권한 기반 CP-ABE 를 블록체인 형태에 맞게 사용한 것이다. 속성 권한을 나누어 관리하기 때문에 두 기법 모두 단일 실패 지점에 대한 위협이 낮다. 그리고 시스템에 존재하는 권한에 대하여 속성 비밀 키 계산을 분산하여 수행하기 때문에, 과부하의 문제도 해결할 수 있다.

일반적인 분산 CP-ABE 와 의 다른 점에 대해서 논하자면 다음과 같다. 공개 파라미터 값 *PP* 에 대하여 PBFT 와 같은 합의 과정을 거친 후 결정하기 때문에, 분산된 권한 노드 중 일부의 악의적인 행위를 방지할 수 있다. 또한 블록체인에 속성 키를 사용할 때 생성한 *PP* 값을 공개하기 때문에 사용자들로 하여금 마스터 키 및 공개 키에 대한 검증을 할 수 있도록 하게 한다. 각 속성 권한이 어떠한 속성을 관리하는 지에 대해서는 블록체인을 통해 투명하게 공개한다.

제안 모델에서 CP-ABE 가 가지는 의미는 다음과 같다. 우선 OSP 조직이 권한을 CP-ABE 의 분산된 권한의 역할을 수행한다. 이들은 관리하고 있는 각 사용자의 계정에 대한 관리함과 동시에 이를 이용하여 속성 비밀 키를 발급한다.

따라서 내부의 암호화를 위한 시스템만 구축하면 되기때문에, 별도의 속성 권한을 구축하지 않아도 된다. [표 3]은 두 CP-ABE 기법에 대해 비교 분석 한 것이다.

[표 3] 분산 권한 CP-ABE 와 블록체인 기반 CP-ABE

	Multi-Authority CP-ABE [28]	Blockchain based CP-ABE
AA (Attribute Authority)	다중 권한	다중 권한
내부 위협 요소	CA (Central Authority)또는 일부 악의적인 분산 AA	일부 악의적인 분산 AA (전체 1/3 까지 장애 허용)
속성 비밀키 발급	CA	n 개의 AA 들이 합의함으로써 *MSK*, *PK* 생성 속성 비밀 키 발급

V. 결론

본 논문에서는 마이데이터를 이용한 블록체인 기반 저작권 관리 모델을 제안하였다. 제안 모델을 통해 각 온라인 서비스 플랫폼에 분산되어 유통되고 있는 자신의 저작물을 마이데이터 계정과 이용 허락 동의를 기반으로 저작권 및 저작물 공유에 관한 모든 프로세스를 관리할 수 있을 것 이다. 이는 우선 기존의 중앙 집중형 구조의 저작권 관리모델과는 다르게 블록체인 기술을 이용하여 데이터 관리 주체를 탈중앙화 시킴으로써 저작권 관리에 대한 투명성을 재고한다. 또한 지금까지 연구 및 개발되었던 블록체인 기반의 저작권 관리 모델이 제안하였던 블록체인 기반 단일 플랫폼과도 차별화 되어있다.

본 논문에서 제안한 마이데이터 개념을 이용한 저작권 관리 모델이 시사하는 바는 다음과 같다. 창작자들이 생산해내는 저작물 데이터의 양은 굉장히 많아지고 그것을 다루는 온라인 서비스 플랫폼들이 많아지는데, 저작권자는 그러한 플랫폼들에 분산되어 있는 자신들의 데이터 관리를 일일이 할 수 가 없으며 그런 점을 악용한 저작권 침해사고가 발생하였다. 제안 모델은 플랫폼들간의 블록체인 네트워크 형성 후 그를 기반으로 하는 마이데이터 계정을 이용하여 저작권자가 직접 데이터 사용 및 이용 동의 내용을 직접 설계함으로써 네트워크 내에 배포된 자신의 데이터를 관리할 수 있다. 그리고 사용자들에게 플랫폼을 대여하는 온라인 서비스 사업자들은 이들로 하여금 창작물을 재생, 게시 및 사용하도록 해준다. 이러한 플랫폼을 사용하게 함으로써 저작권자들의 권리를 보호하고 창작을 도모하여 저작물 산업계의 더 큰 가치를 창출하고 이들이 생성하는 창작물을 서비스하기 위해 더 개선된 플랫폼을 연구하고 개발할 수 있을것으로 기대된다.

향후 모든 데이터 사용 흐름은 점차적으로 기업 또는 조직 중심의 관리가 아닌, 개인 중심으로 수행될 것이다. 개인이 생성해내는 모든 데이터들 중 하나로써 저작권 데이터도 마이데이터를 이용한 관리 모델이 개발된다면 조직들 사이에서 저작권자 중심의 저작권 관리 모델을 충실히 수행할 수 있을 것으로 기대된다.

참고문헌

[1] 한국저작권위원회 - 저작물, [Online], Available : https://www.copyright.or.kr/information-materials/common-sense/knowledge-for-netizen/index.do

[2] Wikipedia- - Copyright, [Online], Available : https://en.wikipedia.org/wiki/Copyright

[3] 한국저작권위원회, 블록체인 기술을 활용한 저작권 신 서비스 연구, 저작권정책연구 2017-6 호, 2017 년 6 월.

[4] Savelyev, Alexander. "Copyright in the blockchain era: Promises and challenges." *Computer law & security review* 34.3 (2018): 550-561.

[5] 신탁관리단체,[Online], Available : https://www.gokams.or.kr:442/artsdb/05_report/copyright05.asp

[6] Poikola, Antti, Kai Kuikkaniemi, and Harri Honko. "Mydata a nordic model for human-centered personal data management and processing." *Finnish Ministry of Transport and Communications* (2015).

[7] MyData Service, Docs,[online] : Avaliable : https://hiit.github.io/mydata-stack/

[8] 한국신용정보원, “해외 마이데이터 사례 분석 및 국내 적용을 위한 시사점 도출”, CIS 이슈리포트 2018-11 호, 2018

[9] Zheng, Zibin, et al. "An overview of blockchain technology: Architecture, consensus, and future trends." Big Data (BigData Congress), 2017 IEEE International Congress on. IEEE, 2017.

[10] Sankar, Lakshmi Siva, M. Sindhu, and M. Sethumadhavan. "Survey of consensus protocols on blockchain applications." Advanced Computing and Communication Systems (ICACCS), 2017 4th International Conference on. IEEE, 2017.

[11] Andreas M. Antonopolous, (2018).The blockchain ,*Mastering Bitcoin*, pp 202-203.

[12] Buterin, Vitalik. "A Next-Generation Smart Contract and Decentralized Application Platform, Ethereum White Paper." 2014.

[13] Merkle Patricia Tree, [Online], Available : https://ethereum.stackexchange.com/questions/6415/eli5-how-does-a-merkle-patricia-trie-tree-work

[14] Bethencourt, John, Amit Sahai, and Brent Waters. "Ciphertext-policy attribute-based encryption." *2007 IEEE symposium on security and privacy (SP'07)*. IEEE, 2007.

[15] Zhang, Yichen, JiguoLi, and HaoYan. "Constant Size Ciphertext Distributed CP-ABE Scheme With Privacy Protection and Fully Hiding Access Structure." IEEE Access 7 (2019): 47982-47990.

[16] My Data Docs, [Online]. Available : https://github.com/HIIT/mydata-service-linking.pdf

[17] Castro, Miguel, and Barbara Liskov. "Practical Byzantine fault tolerance." *OSDI*. Vol. 99. No. 1999. 1999.Dominique Guegan,

[18] David LEE Kuo Chuen, Robert H. Deng. “Blockchain - From Public to Private”, Handbook of Blockchain Digital Finance and inclusion , pp. 145-177, 2017.

[19] My Data Authz Docs, [Online]. Available: https://github.com/HIIT/mydata-stack/raw/gh-pages/mydata-data-authz.pdf

[20] Github - Hyperledger SHIM , [Online], Available : https://github.com/hyperledger/fabric/core/chaincode/shim/

[21] Github - DRM.go [Online]. Avaiable : https://github.com/mitmedialab/2019-MIT-Computational-Law-Course/blob/master/IBM-Blockchain-DRM/drm.go#L97

[22] Hyperledger Fabric Docs - Chaincode, [Online], Available : https://hyperledger-fabric.readthedocs.io/en/release-1.4/smartcontract.html

[23] IBM Developers - IBP, [Online]. Avaiable : https://developer.ibm.com/kr/developer-%ea%b8%b0%ec%88%a0-%ed%8f%ac%eb%9f%bc/2019/06/08/ibm-blockchain-platform-extension-for-vs-code-03/

[24] Hyperledger Fabric Docs, Ledger, [Online]. Available : https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html#world-state-database-options

[25] 시미즈 토모노리, 『하이퍼레저 패브릭 철저 입문』.양현 (역). 위키북스, pp136-191. 2018.

[26] Hyperledger Fabric Docs, “Chaincode”,[Online]. Available : https://hyperledger-fabric.readthedocs.io/en/release-1.4/smartcontract.html

[27] Faber, Benedict, et al. "BPDIMS: A blockchain-based personal data and identity management system." *Proceedings of the 52nd Hawaii International Conference on System Sciences*. 2019.

[28] Liu, Zechao, et al. "Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating." *Journal of Network and Computer Applications* 108 (2018): 112-123.