블록체인의 비트코인 유사 트랜잭션을 활용한 선택된 개인정보의 사용자 소유권 입증

(Attestation on User's Ownership of Selected Personal Information using Bitcoin-like Transactions on a Blockchain)

박준철*

(Jun-Cheol Park)

요 약

포털이나 웹사이트의 서버 데이터베이스가 탈취당해 개인정보가 유출되는 사례가 발생할 때마다 서버 보안성 제고에 관심이 집중되지만, 웹사이트마다 회원 개인정보를 저장하는 구조가 변하지 않는 한 근원적인 문제 해결은 어렵다. 본 논문은 비트코인 유사 트랜잭션을 채택하는 블록체인의 트랜잭션 출력 이어감을 통해 사용자가 서비스 제공자가 요구하는 필수 개인정보 요소만을 선택적으로 입증받는 기법을 제시한다. 이때 서비스 제공자의 인증 서버는 어떤 사용자 정보도 보유하지 않으면서도 제시된 정보가 올바르며 제시자가 정보의 소유자임을 확인할 수 있다. 또한 개인정보 입증 과정에서 정보 요소값의 기밀성이 보장되며, 입증 내역은 블록체인에 트랜잭션으로 기록되어 영구히 보존되며 추후 언제든 확인할 수 있다.

■ 중심어 : 입증 ; 블록체인 ; 트랜잭션 ; 비트코인 ; 보안

Abstract

Whenever the databases of portals or websites are breached and personal information is leaked, attention turns to strengthening server security. However, as long as websites continue storing members' personal information on its own, it is difficult to resolve the problem at its root. This paper proposes a method that leverages a chaining of blockchain transaction outputs, adopting Bitcoin-like transactions, to allow users to selectively prove only the essential personal information elements required by a service provider. The service provider's authentication server retains no user information, yet can still verify that the presented information is correct and that the presenter is the legitimate owner of that information. Furthermore, during the verification process, the confidentiality of each information element's value is guaranteed, and the verification records are permanently stored as transactions on the blockchain. These records can be checked at any time in the future.

■ keywords : attestation ; blockchain ; transaction ; Bitcoin ; security

Ⅰ. 서 론

최근의 SKT, 롯데카드, KT 해킹 사례에서 드러났듯이, 통신사나 웹사이트 서버의 개인정보 데이터베이스 탈취는 꾸준히 발생하고 있다. 서비스 제공자는 이름, 성별, 생년월일, 전화번호, 주소 등 회원의 주요 개인정보를 서버에 보유하기에 이런 서버가 공격당해 저장 내용이 유출되면, 가입자는 잘못이 없음에도 개인정보 유출로 인한 피해를 보게 된다. 서비스 제공사는 회원에게 제공한 서비스 기록(예: 구매 내역)을 통해 맞춤형 추천 등을 하거나 만약의 사용자와의 분쟁에 대비할 수도 있기에 사용자 개인정보의 자체 보유를 선호한다. 하지만 개인정보 유, 노출 및 2

* 종신회원, 홍익대학교 컴퓨터공학과
이 논문은 2024학년도 홍익대학교 학술연구진흥비에 의하여 지원되었음.
접수일자 : 2025년 10월 13일
수정일자 : 2025년 11월 13일
게재확정일 : 2025년 11월 24일
교신저자 : 박준철 e-mail : jcpark@hongik.ac.kr

차 피해로 인한 가입자의 경제적, 정신적, 사회적 손실과 파장은 서비스 편의성보다 보안성이 훨씬 더 중요하게 취급될 당위성을 보여준다.

본 논문은 서비스 제공자가 어떤 사용자 개인정보도 저장할 필요 없이 사용자가 제시하는 정보를 Bitcoin(이하 비트코인)[1] 유사 트랜잭션을 통해 유효성을 검증하고 블록체인에 기록을 남기는 방식을 제안한다. 이 방식에서는 사용자가 서비스에 따라 요구되는 최소 요소(예: 여론조사 응답자가 50대 여성인지를 확인하려면, 생년월일과 성별)만을 선별 제출할 수 있으며, 수정된 요소(예: 전화번호)값을 블록체인에 갱신하여 이후 변경된 값을 검증하게 할 수 있고, 필요시 자신의 전체 개인정보 요소(들)를 무효화시켜 위조 공격을 차단할 수도 있다. 제안 블록체인(이하 ALChain: Attestation & Logging Blockchain)은 읽기는 일반에게 공개하되, 쓰기는 지정된 참여자들에게만 허용하는 신뢰 기관(예: 행정안전부) 운영 하의 사적(private) 체인이다.

본 논문의 구성은 다음과 같다. 2장에서 서버 데이터베이스 대용의 블록체인 활용 사례들을 제안 기법과 비교하고, 3장에선 비트코인 블록체인 대비 ALChain의 차이를 트랜잭션 중심으로 서술한다. 4장에서 제안 기법의 등록, 입증, 변경, 폐기 프로토콜 설명 후, 5장 및 6장에선 제안 기법의 보안성 및 핵심 요소인 입증 서비스 처리 시간을 각각 분석하고, 7장에서 결론을 맺는다.

Ⅱ. 관련 연구

서버 데이터베이스를 대신하려는 시도의 블록체인 활용 관련 기존 연구 결과들을 분석한다.

1. 블록체인을 인증 수단으로 활용

블록체인 트랜잭션을 통해 사용자가 한 서비스 제공자에게 등록(인증됨)한 후, 다른 서비스 사용자에게 추가 등록 없이 인증받기 위해 블록체인 스마트 계약(contract) 내의 사용자 식별자 및 크리덴셜을 활용하는 기법[2]이 발표되었다. 이때 스마트 계약을 통해 사용자의 만료된 권한 행사를 막는 등 동적인 접근 제어가 가능하다. 다만 저장하는 사용자 정보는 인증 또는 불인증 판정을 위한 목적을 가져, 제안 기법은 선택적 정보 요소를 인증 또는 자격 검증의 목적으로 사용할 수 있어 활용 범위가 더 넓고 스마트 계약을 쓰지 않아 수수료 부담이 없다.

X.509 표준 대신 스스로 서명한 가벼운(크기 적음) 인증서를 Ethereum(이더리움) 블록체인에 등록하여 모바일 단말기나 IoT 기기를 인증하는 기법[3]이 제시되었다. 인증서 공개키의 유효성은 블록체인 변조 저항성으로 보장한다. 그렇지만 사용자 아닌 기기를, 공개키 암호화 방식으로 인증한다는 점에서 제안 기법과 차별된다.

블록체인을 활용하여 2FA(Two-factor Authentication)를 달성하려는 연구[4]가 있었다. 사용자는 MetaMask 지갑의 이더리움 계정을 통해 지정된 스마트 계약에 접근하는 트랜잭션을 발생시켜 인증을 시도한다. 이를 통해 웹 서비스 접근을 요청한 사용자가 승인받을 수 있다. 하지만 두 번째 인증 요소인 블록체인이 자격 검증에는 쓰이기 어렵고, 스마트 계약 실행의 수수료가 발생한다는 점에서 제안 기법과 차별된다.

PUF(Physical Unclonable Function)를 이중 인증 요소로 채택하여 PUF 칩이 탑재된 의료용 센서를 부착한 환자와 모니터링 센서를 관리하는 의료진 사이의 인증 및 데이터 교환 기법[5]이 제안되었다. 스마트 계약을 지원하는 블록체인 트랜잭션으로 인증 요소 및 의료 데이터에의 접근을 관리하는데, 인증에서 센서의 PUF 고유성을 활용하는 특징을 지닌다. 블록체인이 기기 인증과 의료 데이터의 저장 및 접근 제어에 활용되기에, 사용자 인증 및 자격 검증 목적의 제안 기법 활용 방식과는 차이가 있다.

제안 기법처럼 블록체인을 활용해 서버의 개인정보 저장 없이도 필요한 정보만을 서비스 제공

자에 제시하는 기법[6]이 발표되었다. 다만 [6]의 기법은 정보의 유효성 입증 시에 서비스 제공자 서버 외의 신뢰 기관의 개입이 필요하며, 사용자 정보 중 바뀔 수 없는 고정된 값(들)의 조합만으로 자격 증명을 제공한다는 한계를 지닌다.

2. 블록체인을 보안 데이터베이스로 활용

여러 의료기관 간의 의료정보의 공유를 위해 블록체인을 활용하는 기법[7]이 제안되었다. 블록체인을 통해 한 병원의 환자 처치 내용을 다른 병원에 공유하며, 저장된 데이터에 대해 패스워드 및 생체인식 기반의 접근 제어를 제공한다. 다만 연구 목적이 의료 데이터의 안전하며 가용성 있는 관리이기에 제안 기법과 차이가 있다.

의료정보의 블록체인 저장 시 시스템의 확장성을 고려해 별도의 사이드체인을 두는 방식[8]이 제안되었다. 즉 방대한 양의 의료정보를 메인 체인이 아닌 별도 사이드체인에 저장하고, 이에 접근하는 의료진 등을 인증할 때 스마트 계약에서 접근 제어를 담당하게 함으로써 데이터 훼손이나 유출을 억제하려는 것이다. 그렇지만 [8]의 결과는 블록체인을 보안성이 있는 단순 분산 데이터베이스로 활용하기에, 블록체인을 인증 수단으로 삼아 활용하는 제안 기법과는 차이가 있다.

Ⅲ. ALChain 블록체인의 비트코인 블록체인 대비 차별성 및 설계 근거

비트코인 대비, 트랜잭션 구성과 실행 방식, 블록의 생성과 블록체인 관련한 ALChain의 차이와 그 설계 이유를 설명한다. ALChain을 활용하는 제안 기법 참여자들은 등록 기관 서버(이하 registrarS: Registrar Server), 다수의 사용자 및 여러 서비스 제공자의 인증 서버들이다.

1. 트랜잭션의 Input과 Output

비트코인 트랜잭션은 하나 이상의 input 및 output으로 구성되며, input(들)에서 이미 존재하는 output(들)으로 전송된 BTC(비트코인 시스템의 화폐)를 사용하여 새로운 output(들)으로 BTC를 전송(소비)하는 역할을 한다. 예로, 그림 1의 첫 번째 화살표는 중간 트랜잭션의 input이 맨 위 트랜잭션의 네모가 가리키는 output(인덱스 번호 #0)을 참조하여(가져다가) 사용함을 보여준다. 그 결과 중간(두 번째) 트랜잭션의 output들(인덱스 번호 #0는 Bob의 주소로 0.0150 BTC 전송, 인덱스 번호 #1은 Alice의 (또 다른) 주소로 0.0845 BTC 전송)이 생긴다. 트랜잭션 식별자는 트랜잭션에 해시 함수를 적용한 결과이며 콜론(:) 다음의 수는 인덱스 번호이다.

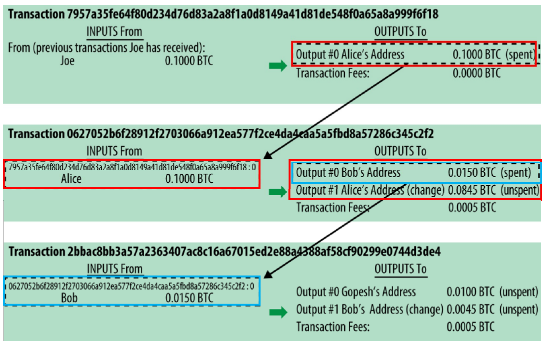


그림 1. 비트코인 트랜잭션의 output 참조하기 예[9]

Input-Output 기준 트랜잭션 형태는 그림 2(a)처럼 한 output을 가져와 목적지(output #0)로 보내고 남은 금액을 회수(output #1)하는 식이 보편적이나 2(b)-2(d)와 같은 형태도 가능하다.

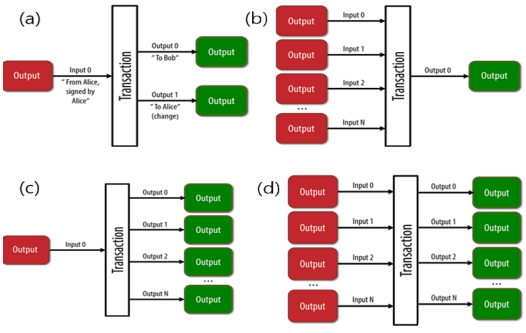


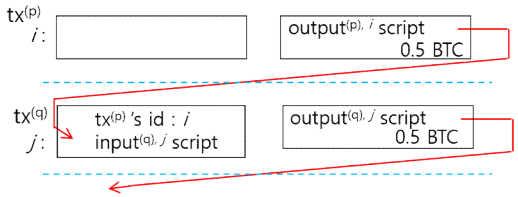
그림 2. input-output 기반 비트코인 트랜잭션 유형[9]

한편, ALChain의 트랜잭션도 input 및 output으로 구성되지만 그림 2(d)처럼 항상 같은 수의 input과 output을 가진다. 각 input-output 쌍은

개인정보 요소 하나에 대한 입증을 이어가는 역할을 하면서, 같은 트랜잭션의 다른 input-output 쌍(들)과는 같은 소유자의 개인정보 요소(들)라는 점을 제외하고는 각각 독립적이다. 또한 ALChain 내의 어떤 통용 화폐도 필요치 않다. 더불어 모든 종류의 트랜잭션은 각기 정해진 역할만을 담당하기에 그 input 및 output의 형태가 고정되어 있고 그 형태로만 사용된다.

2. 트랜잭션의 실행: 스크립트와 고정 절차

그림 3과 같이 output에는 locking 스크립트가 있어 이를 풀어내는(unlocking) 스크립트를 가진 input에서 이 output을 사용할(가져올) 수 있다. Locking의 종류에는 공개키의 해시값을 지정하거나(P2PKH: Pay-to-PublicKeyHash), 스크립트 해시값을 지정하는 것(P2SH: Pay-to-Script Hash) 등이 있다. 그림 3은 $tx^{(p)}:i$의 output에 들어있는 0.5 BTC를 가져와서 $tx^{(q)}$의 인덱스 j의 input에서 사용하는 것을 도식화한 것이다. 이를 위해 $tx^{(q)}:j$의 input script는 $tx^{(p)}:i$ output script에 연결되어 실행됨에 문제가 없어야 한다. 참조가 가능한 output을 Unspent Transaction Output(이하 UTXO)라 하는데, 그림 3은 참조된(화살표 머리) UTXO가 사용되고 새로운(화살표 꼬리) UTXO가 생김을 나타낸다.



(i) $tx^{(q)}$ 의 input$^{(q),j}$ script가 $tx^{(p)}$ 의 output$^{(p),i}$ script ($tx^{(p)}$'s id : i와 같이 transaction id 및 index를 지정)를 unlock 함;
(ii) $tx^{(q)}$ 의 output$^{(q),j}$ scrip는 다시 이후 transaction이 이를 unlock 하는 input을 제시함으로써 UTXO 사용을 이어 나감;

그림 3. 비트코인 트랜잭션 스크립트 실행 이어짐

이에 비하여, ALChain 트랜잭션은 처리 시 모든 참여자는 항상 정해진 절차를 따르므로 input이나 output에 따로 스크립트 코드를 넣지 않는다. ALChain의 트랜잭션은 하나 이상의 input-output 쌍(인덱스 번호로 구분)으로 구성된다. Input은 콜론(:)으로 구분되어 앞부분은 트랜잭션 식별자, 뒷부분은 이 식별자의 트랜잭션 내의 특정 input-output 쌍을 지칭하는 인덱스 번호(0부터 시작)이다. 또한 사용자가 개인정보 요소(들)를 registrarS를 통해 최초 ALChain에 등록할 때(즉 이어갈 이전 output(들)이 없을 때)는 input이 null:null 형태가 된다. Output도 콜론(:)으로 구분되는데 앞부분은 한 개인정보 요소의 이중 해시 적용 결과, 뒷부분은 이 요소를 참조할 때 필요한 공개키의 해시 적용 값이다. 즉, output이 randStr1: randStr2 라면, randStr1은 해당 요소값 $elmt$을 찾으려는 모든 가능성의 무차별 대입 공격에 대비하여 충분히 긴 1,024비트의 일회용 랜덤 값 $rpad$를 포함하는 $h_2(h_1(elmt\|rpad)\|rpad\|elmt)$로 놓고, randStr2는 서명 검증용 공개키 $pubKey$에 해시 적용한 $h_1(pubKey)$로 놓는다.

그림 4의 첫 번째 input-output 쌍은 트랜잭션 tx_p의 인덱스 번호 a에 해당하는 쌍으로서, 이것을 트랜잭션 tx_q의 인덱스 번호 b에서 참조하고 있다. 그래서 $tx_q:b$(트랜잭션 식별자:인덱스 번호) 쌍의 input은 $tx_p:a$가 된다. 이때 $tx_q:b$ 쌍의 output은 트랜잭션 $tx_p:a$의 output에서 지정한 개인정보 요소 $hElmt_x$을 그대로 포함하고 있어, ①은 요소 $hElmt_x$를 참조해 소유자임을 입증한 후, 이 요소를 이후의 참조에서 다시 사용하는 흐름을 보인다. 반면 ②는 소유자가 이 요소의 새로 갱신된 값 $hElmt_{x*}$(예: 새 주소)를 등록하는 흐름으로, 기존 $hElmt_x$은 사용 불가가 된다.

한 사용자의 여러 개인정보 요소는 참조된 횟수에 따라 그림 4의 유형 ①의 흐름에 의해 연결된 input-output 쌍의 개수가 달라진다. 예로, 홍길동의 전화번호는 10회, 생년월일은 7회 참조(입증)되었다면, 최초 등록 트랜잭션 이후로 전화번호는 10회, 생년월일은 7회의 유형 ① 연결이 각각 이어진 상태가 된다. 이하 사용자 U_i의 개인정보 요소마다 가장 최근 사용된(또는 등록

된) 트랜잭션의 인덱스 번호에 해당하는 output의 집합을 $UTXO_{U_i}$라 한다. 그림 5는 한 사용자가 입증 트랜잭션 tx_{NEW}를 생성할 때 이전 tx_Z로부터 "휴대폰 번호"와 "이름"을, tx_Y로부터 "생년월일"을, tx_X로부터 "이메일 주소"를 인덱스 순서대로 이어감과 그 결과에 따른 UTXO의 갱신을 보여준다.

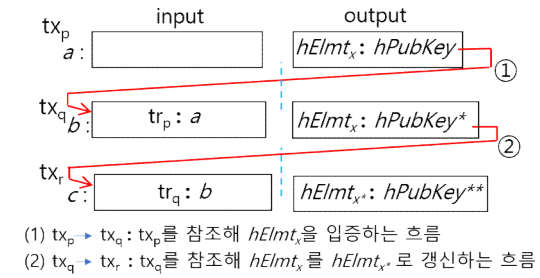


그림 4. ALChain의 트랜잭션 참조 흐름의 유형

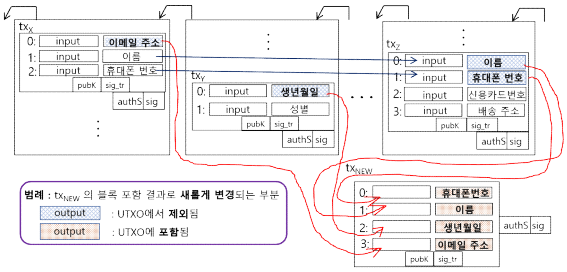


그림 5. ALChain의 한 사용자 UTXO 집합의 갱신 예

3. 블록의 생성 및 블록체인 이어감

비트코인 채굴자(miner)는 비트코인 시스템의 주요 참여자로, 사용자 트랜잭션의 유효성을 검증 후 트랜잭션들을 포함하는 블록을 구성하고 채굴용 해시 퍼즐을 풀고자 시도한다. 채굴 성공자는 새로 생성된 BTC를 채굴 보상으로 받고, 채굴된 블록은 비트코인 블록체인에 이어지게 된다. 가장 긴 길이의 블록체인이 다수에게 인정받는 체인이 되며, 이런 채굴 경쟁이 평균 10분의 블록 생성 주기마다 반복된다.

한편, 비트코인에서와 마찬가지로 ALChain의 블록은 그림 5에 보이듯이 트랜잭션(들)을 포함하는 단위로서 해시 포인터로 이어지는 체인의 구성 요소가 된다. 이런 ALChain 블록을 어떻게 구성하고, 누가(지분 증명(PoS: Proof of Stake) 방식의 검증자(validator) 또는 작업 증명 (PoW: Proof of Work) 방식의 채굴자 등, 다만 블록 생성에 경쟁 자체가 없기에 PoW 방식 채굴 필요성 사라짐), 얼마나 자주(예: 매 3초) 제안하며, ALChain에 어떻게 체인화(이을지)할지 등은, 블록이 제안 기법의 트랜잭션 output 이어감을 지원하는 한 제안 기법 실현에 영향을 주지 않는다. 이때 ALChain에는 허가받은 노드(검증자 등)들만이 쓰기를 할 수 있기에 제안(생성)된 블록은 항상 유효하며 다른 노드 역시 당연히 유효하다고 인정할 것임을 가정해도 좋다.

Ⅳ. 초기 등록, 입증 및 철회 프로토콜

ALChain을 활용해 사용자가 개인정보 요소(들)의 유효성을 서비스 제공자 서버에게 입증하는 절차와 이를 위해 사전에 사용자 개인정보 요소(들)를 registrarS에 등록하는 절차, 등록된 요소의 변경 및 폐기를 위한 과정의 프로토콜을 각각 서술한다. 이하 모든 프로토콜의 교환되는 메시지는 전부 TLS 1.3으로 보호되며, 연접 연산자를 ||, 크립토 해시 함수 SHA3-512[10]를 h_1, BLAKE-512[11]를 h_2로 쓴다. BLAKE는 2012년 발표된 NIST의 SHA-3 패밀리 공모전의 최종 5개 알고리즘 중 하나이다.

1. 사용자 개인정보 초기 블록체인 등록

사용자의 개인정보 요소(들)를 최초로 ALChain 트랜잭션으로 등록하는 절차는 프로토콜 1과 같다.

프로토콜 1 : 사용자 U_i(스마트폰 M_i) 개인정보 요소(들) registrarS에 인증 및 ALChain 트랜잭션으로 등록

1. $U_i\,(M_i) \leftrightarrow registrarS$: 상호 인증
2. $U_i\,(M_i) \rightarrow registrarS$: 트랜잭션 $tx_{registration}$ 및 등록할 개인정보 요소(들)

// $tx_{registration}$: $s(\geq 1)$개 input-output 쌍으로 구성

$(null : null, f(elmt_0, rpad_0^0) : hPubKey_0^0)$

$(null : null, f(elmt_1, rpad_1^0) : hPubKey_1^0)$

. . . . . .

$(null : null, f(elmt_{s-1}, rpad_{s-1}^0) : hPubKey_{s-1}^0)$

// 등록할 개인정보 요소(들)의 구성:

$\forall j\,(j = 0,1, \ldots, s-1)$에 대하여:

$elmt_j$, $rpad_j^0$

3. $registrarS$: 개인정보 요소(들) 검증 진행;

$\forall j\,(j = 0,1, \ldots, s-1)$에 대하여,

$elmt_j$가 U_i의 해당 개인정보 요소임을 확인

(예: 행정안전부 데이터베이스 기록과 대조);

$h_2(h_1(elmt_j \| rpad_j^0) \| rpad_j^0 \| elmt_j)$

$== f(elmt_j, rpad_j^0)$ 인지 확인;

4. $registrarS$: $tx_{registration}$ 이용해 블록에 포함될 확장 트랜잭션 $tx_{registration}^{confirmed, registrarS}$를 다음과 같이 생성;

$tx_{registration} \| registrarS(URI) \| sig_{registrarS}$

($sig_{registrarS}$ 는 $tx_{registration}$ 검증했음을 블록에 남기기 위해 $registrarS$가 $tx_{registration}$에 한 서명)

==========

사용자 U_i의 최초 등록 $tx_{registration}$ 트랜잭션은 개인정보 각 요소를 하나의 output에 매치한다. 그러면 registrarS가 수신한 올바른 요소값(들)과 $rpad$가 각각 이후의 참조에서 제시되어야 이 output(들)을 입증할 수 있다. 트랜잭션 $tx_{registration}^{confirmed, registrarS}$의 식별자가 $txid_{reg,U_i}$일 때 등록 후 U_i의 최초 $UTXO_{U_i}$는 {$txid_{reg,U_i}$: 0의 output , ... , $txid_{reg,U_i}$:$s-1$의 output}이 된다.

2. 인증 서버에 의한 개인정보 입증과 기록

사용자가 개인정보 요소(들)를 서비스 제공자의 인증 서버에 입증하면서 내역을 ALChain 트랜잭션으로 기록하는 절차는 프로토콜 2와 같다.

프로토콜 2 : 서비스 요청을 위한 사용자 U_i(스마트폰 M_i)의 최소 개인정보 요소(들) $authS_k$에 입증 및 ALChain에 입증 사실 기록한 트랜잭션 등록

==========

1. $U_i\,(M_i) \rightarrow authS_k$: 특정 서비스 요청
2. $U_i\,(M_i) \leftarrow authS_k$: 서비스 제공을 위해 필요한 최소 개인정보 요소(들)
3. $U_i\,(M_i) \rightarrow authS_k$: 트랜잭션 $tx_{attestation}$ 및 최소 개인정보 요소(들)의 입증 정보

// $tx_{attestation}$: $t(\geq 1)$개 input-output 쌍 및

// 서명 검증용 공개 키와 서명으로 구성

$(txid_l : idx_l,$

$f(elmt_l, rpad_l^{c(l)+1}) : hPubKey_l^{c(l)+1})$

$(txid_m : idx_m,$

$f(elmt_m, rpad_m^{c(m)+1}) : hPubKey_m^{c(m)+1})$

. . . . . .

$(txid_n : idx_n,$

$f(elmt_n, rpad_n^{c(n)+1}) : hPubKey_n^{c(n)+1})$

$pubKey_n^{c(n)}$; $sig_{attestation}$

// ($sig_{attestation}$는 $pubKey_n^{c(n)}$의 쌍인 개인 키로

// $tx_{attestation}$에 서명한 것. 따라서 $txid_n : idx_n$

// 의 ouput 뒷부분 $hPubKey_n^{c(n)}$을 해시 값으로

// 가지는 $pubKey_n^{c(n)}$를 사용해 검증할 수 있음)

// (단, 각 $j = l,m, \ldots, n$에 대해, $txid_j : idx_j$의

// output은 $f(elmt_j, rpad_j^{c(j)}) : hPubKey_j^{c(j)}$이며,

// $c(j)$는 $elmt_j$가 현재까지 참조(입증)된 횟수)

// 최소 개인정보 요소(들)의 입증 정보:

$\forall j\,(j = l,m, \ldots, n)$에 대하여:

$elmt_j$, $rpad_j^{c(j)}$, $rpad_j^{c(j)+1}$

4. $authS_k$: 다음의 개인정보 요소(들) 입증 진행;

4.1 각 $txid_j : idx_j\,(j = l,m, \ldots, n)$의 output이 전부 $UTXO_{U_i}$에 속하는 사용자 U_i가 존재하는지 확인; // 동일 사용자의 개인정보 요소들임을 검증

4.2 각 $j = l,m, \ldots, n$에 대하여,

$h_2(h_1(elmt_j \| rpad_j^{c(j)}) \| rpad_j^{c(j)} \| elmt_j)$

$== f(elmt_j, rpad_j^{c(j)})$ 인지 확인;

$h_2(h_1(elmt_j \| rpad_j^{c(j)+1}) \| rpad_j^{c(j)+1} \| elmt_j)$

$== f(elmt_j, rpad_j^{c(j)+1})$ 인지 확인;

// 같은 $elmt_j$ 값이 이어지도록 구성되었음을 검증

4.3 $h_1(pubKey_n^{c(n)}) == hPubKey_n^{c(n)}$ 인지 확인;

4.4 서명 $sig_{attestation}$이 유효한지를 서명 검증 공개키 $pubKey_n^{c(n)}$ 및 $tx_{attestation}$를 이용하여 확인;

5. $authS_k$: $tx_{attestation}$ 이용해 블록에 포함될 확장 트랜잭션 $tx_{attestation}^{confirmed,authS_k}$를 다음과 같이 생성;

$tx_{atestation} || authS_k(URI) || sig_{attestation}^{authS_k}$

($sig_{attestation}^{authS_k}$는 $tx_{attestation}$ 검증했음을 블록에 남기기 위해 $authS_k$가 $tx_{attestation}$에 한 서명)

==========

사용자 U_i는 개인정보 요소(들) 각각에 해당하는 output(UTXO)을 참조하는 트랜잭션을 위해, 각 $rpad$와 요소값, 서명 검증용 공개키와 서명을 제시한다. 인증 서버는 참조된 output 전부가 $UTXO_{U_i}$에 속하며(4.1), 각 $rpad$ 쌍을 이용해 트랜잭션의 참조된 output 및 새로운 output(이후 참조될)의 요소값이 모두 올바르며(4.2), 공개키의 해시 결과를 비교하고(4.3), 제시된 서명이 트랜잭션의 서명임을 공개키로 확인한다(4.4).

3. 사용자 개인정보 일부 요소(들) 값 갱신

개인정보 요소(들)의 새로운 값을 ALChain 트랜잭션으로 기록하는 절차는 프로토콜 3과 같다.

프로토콜 3 : 사용자 U_i(스마트폰 M_i)의 일부 개인정보 요소(들)의 갱신 값 registrarS에 인증 및 ALChain에 갱신된 요소(들) 반영한 트랜잭션 등록

==========

1. $U_i\,(M_i) \rightarrow registrarS$: 트랜잭션 tx_{update} 및 갱신할 개인정보 요소(들) 입증 정보

// tx_{update}: $t(\geq 1)$개의 input-output 쌍 및
// 서명 검증용 공개키와 서명으로 구성

$(txid_l : idx_l, f(elmt_{l*}, rpad_{l*}^0) : hPubKey_{l*}^0)$

$(txid_m : idx_m, f(elmt_{m*}, rpad_{m*}^0) : hPubKey_{m*}^0)$

. . . . . .

$(txid_n : idx_n, f(elmt_{n*}, rpad_{n*}^0) : hPubKey_{n*}^0)$

$pubKey_n^{c(n)}$; sig_{update}

// (단, 각 $j = l, m, \ldots, n$에 대해, $txid_j : idx_j$ 의
// output은 $f(elmt_j, rpad_j^{c(j)}) : hPubKey_j^{c(j)}$이며,
// $c(j)$는 $elmt_j$가 현재까지 참조(입증)된 횟수)
// 갱신할 개인정보 요소(들) 입증 정보:

$\forall j\,(j = l, m, \ldots, n)$에 대하여:

$elmt_j, rpad_j^{c(j)}, elmt_{j*}, rpad_{j*}^0$

// (sig_{update}는 $pubKey_n^{c(n)}$의 쌍이 되는 개인 키로
// tx_{update}에 대해 서명한 것. 따라서 $txid_n : idx_n$
// 의 ouput 뒷부분 $hPubKey_n^{c(n)}$을 해시 값으로
// 가지는 $pubKey_n^{c(n)}$를 사용해 검증할 수 있음)

2. $registrarS$: 갱신 개인정보 요소(들) 입증 진행;

2.1 각 $txid_j : idx_j\,(j = l, m, \ldots, n)$의 output이 전부 $UTXO_{U_i}$에 속하는 사용자 U_i가 존재하는지 확인; // 동일 사용자의 개인정보 요소들임을 검증

2.2 각 $txid_j : idx_j\,(j = l, m, \ldots, n)$의 output이 전부 사용자 변경 가능 요소에 해당하는지 확인;

2.3 각 $j = l, m, \ldots, n$에 대하여, $elmt_{j*}\,(elmt_{j*} \neq elmt_j)$이 올바른지 확인;

2.4. 각 $j = l, m, \ldots, n$에 대하여,

$h_2(h_1(elmt_j || rpad_j^{c(j)}) || rpad_j^{c(j)} || elmt_j) == f(elmt_j, rpad_j^{c(j)})$ 인지 확인;

$h_2(h_1(elmt_{j*} || rpad_{j*}^0) || rpad_{j*}^0 || elmt_{j*}) == f(elmt_{j*}, rpad_{j*}^0)$ 인지 확인;

2.5 $h_1(pubKey_n^{c(n)}) == hPubKey_n^{c(n)}$ 인지 확인;

2.6 서명 sig_{update}이 유효한지 서명 검증 공개키 $pubKey_n^{c(n)}$ 및 tx_{update}를 이용하여 확인;

3. $registrarS$: tx_{update} 이용해 블록에 포함될 확장 트랜잭션 $tx_{uptate}^{confirmed,registrarS}$를 다음과 같이 생성;

$tx_{update} || registrarS(URI) || sig_{update}^{registrarS}$

($sig_{update}^{registrarS}$ 는 tx_{update} 를 검증했음을 블록에 남기기 위해 $registrarS$가 tx_{update}에 한 서명)

==========

사용자 U_i가 개인정보 요소(들) 각각의 기존값과 $rpad$를 제시해 소유자임을 입증하고, 갱신 값 및 $rpad*$ 및 이를 내포하는 output(들)을 가지는 트랜잭션을 제시한다. registrarS는 참조된 output(들) 모두 $UTXO_{U_i}$에 속함(2.1), 변경 가능 요소(들)인지 확인(2.2, 2.3), 기존값 및 새로운 값의 내포(2.4), 공개 키의 해시 함수 결과 비교(2.5), 제시된 서명을 공개키로 검증(2.6)한다.

4. 사용자 등록된 개인정보 요소(들) 폐기

ALChain의 사용자 개인정보 요소(들)의 전체 폐기(사용 불가 만듦) 절차는 프로토콜 4와 같다.

프로토콜 4 : 사용자 U_i의 모든 개인정보 요소(들) 폐기 요청의 registrarS 인증 및 이후 사용 불가토록 ALChain에 트랜잭션 등록

==========

1. $U_i(PC/in\ person) \leftrightarrow registrarS$: 상호 인증 및 U_i의 등록된 개인정보 요소(들) 폐기 요청
2. $registrarS: UTXO_{U_i}$ 검색
   // $UTXO_{U_i}$ 원소인 output을 포함하는 txid:idx
   // $txid_l : idx_l$
   // $txid_m : idx_m$
   // . . . . . .
   // $txid_n : idx_n$
3. $registrarS$: 트랜잭션 $tx_{revocation}$을 생성;
   // $tx_{revocation}$:하나 이상의 input-output 쌍으로 구성
   $(txid_l : idx_l, null : null)$
   $(txid_m : idx_m, null : null)$
   . . . . . .
   $(txid_n : idx_n, null : null)$
4. $registrarS$: $tx_{revocation}$ 이용해 블록에 포함될 확장 트랜잭션 $tx_{revocation}^{confirmed,registrarS}$를 다음과 같이 생성;
   $tx_{revocation} \| registrarS(URI) \| sig_{revocation}^{registrarS}$
   ($sig_{revocation}^{registrarS}$는 $registrarS$가 $tx_{revocation}$에 한 서명)

==========

사용자 U_i의 개인정보 요소(들) 폐기 요청에 대해, registrarS는 input에서 $UTXO_{U_i}$의 output 각각을 전부 참조하며 output이 null:null인 트랜잭션을 생성한다. 그럼 $tx_{revocation}^{confirmed,registrarS}$ 트랜잭션의 블록 포함 후 $UTXO_{U_i} \leftarrow \varnothing$이 되어, 추후 U_i의 개인정보 요소 입증 시도는 원천 거부된다.

V. 보안성 분석

가능한 위험 요소에 대하여 제안 기법이 보안성 측면에서 어떻게 대비되어 있는지 분석한다.

1. 사용자 개인정보 요소 동일성 조작 시도

초기 설정 시 registrarS는 사용자 개인정보 요소(들)값의 유효성 검증 후, 트랜잭션의 각 요소 $elmt$ 해당의 output 앞부분이 $h_2(h_1(elmt \| rpad^0) \| rpad^0 \| elmt)$와 같은지 사용자가 제출한 $rpad^0$를 통해 확인한다. 따라서 이후 이 output을 참조하여 $elmt^*(\neq elmt)$를 제시하면서 서비스 제공자의 인증 서버를 속이려면, $f(elmt^*, rpad^*) = h_2(h_1(elmt^* \| rpad^*) \| rpad^* \| elmt^*)$이 $h_2(h_1(elmt \| rpad^0) \| rpad^0 \| elmt))$과 같아지는 $rpad^*$를 구해서 $elmt^*$와 함께 제시해야 한다. 이는 h_1 및 h_2의 강력한 역상 저항성으로 불가능에 가깝다.

등록 후, 서비스 제공자의 인증 서버는 프로토콜 2의 4.2에 따라, $(txid:idx, f(elmt, rpad_l^{cnt}) : hPubKey^{cnt})$에서 $cnt \geq 1$이면, $f(elmt, rpad^{cnt})$의 "(다음번에)참조될" $elmt$이 $f(elmt, rpad^{cnt-1})$의 "(지금)참조된" $elmt$과 같음을 $rpad^{cnt}$를 써서 확인한다(단, $txid:idx$의 output은 $f(elmt, rpad^{cnt-1}) : hPubKey^{cnt-1}$ 임). 그러므로 이후 반드시 같은 요소값 $elmt$으로 참조가 이어짐이 보장된다. 다만, 수정 허용된 요소가 변경되면 $(txid:idx, f(elmt_{upd}, rpad_{upd}^0) : hPubKey_{upd}^0)$처럼 참조되어 이후는 반드시 $elmt_{upd}(\neq elmt)$를 입증하게 구성된다.

결론적으로, 등록된 값이 아닌 거짓된 개인정보 요소(예: 미성년자가 성인 해당의 생년월일) 제시로 상대를 속이려는 시도는 성공할 수 없다.

2. 사용자 개인정보 요소 탈취 시도

사용자의 개인정보 요소 탈취는 두 가지 경로로 시도될 수 있다. 첫째는 사용자와 서비스 제공 인증 서버 또는 registrarS와의 통신 구간인데, 교환되는 메시지는 TLS 1.3으로 보호되기에 평문 노출의 가능성은 극히 낮다. 둘째, ALChain의 트랜잭션 내용인데, 개인정보 요소 $elmt$는 트

랜잭션 output 앞부분에 $h_2(h_1(elmt \| rpad) \| rpad \| elmt)$와 같이 이중 해시로 감추어져 있다. 이에 설사 추측이 가능한 값이더라도(예: 성별의 '남자'), 충분히 긴 1,024비트의 랜덤 $rpad$가 $elmt$의 기밀성을 보장한다. 왜냐하면 공격자가 $elmt$를 알기 위해서는 $rpad_{trial}$에 0에서 $2^{1024}-1$까지의 값을 차례로 대입하면서 $h_2(h_1(\text{"남자"} \| rpad_{trial}) \| rpad_{trial} \| \text{"남자"})$이 공격 대상 output의 이중 해시값과 같아질 때까지 반복해야 하기 때문이다. 그러므로 성공까지 평균 $2^{1024}/2 = 2^{1023}$의 시도 및 시도마다 h_1과 h_2의 연산이 필요하기에, 추측값 확정에 필요한 해시 함수의 평균 연산 횟수는 $2 \times 2^{1023} = 2^{1024}$이다. 이는 공격자의 시도 의지 자체를 꺾는 매우 큰 값이며, 추측이 불가한 요소에 대해서는 값을 알아내기가 더더욱 어렵게 된다. 실사례로, 비트코인은 블록 채굴자가 앞부분이 충분히 긴 0의 연속인 SHA-256 해시값을 출력하는 입력 일부를 찾기 위해서 후보 값을 무차별 대입하는 것 외에 더 나은 방법이 없음에 작업 증명의 보안성을 전적으로 의지한다.

3. 피해자의 개인정보 요소 위장 입증 시도

공격자가 사용자 U_i로 위장하여 어떤 서비스 제공자를 속이려면, U_i의 개인정보 요소(들)값과 마지막 인덱스 위치에서 참조하는 $UTXO_{U_i}$ output의 뒷부분에 기재된 해시값을 가지는 공개키 및 이 공개키로 검증할 수 있는 서명을 제시해야 한다. 하지만 위 2의 분석에 따라, 공격자의 ① U_i의 개인정보 요소값 얻기 및 ② h_1의 역상을 구해 공개키 얻기 모두가 불가능에 가까우며, 추가로 ECDSA 알고리즘 깨기를 의미하는 ③ 개인키 구하기 또는 서명 생성하기를 달성해야 하므로 이는 무시할 수준의 가능성을 가진다.

4. 사용자 개인정보 입증 서비스 거부 시도

ALChain은 확장 트랜잭션과 블록의 생성, 체인 이어가기 등 쓰기 관련 내용을 허가받은 노드들에만 허용한다. 따라서 막대한 지분이나 다수의 해시 파워 등을 써서 ALChain의 안정성을 훼손하는 공격은 고려 대상이 아니다. 또한 고장 등으로 ALChain 일부 노드가 다운되어도, 다수의 노드가 같은 체인을 유지하므로 입증 서비스 자체가 중단될 가능성은 무시할 정도로 낮다.

VI. 입증 시간 및 트랜잭션 처리량 분석

서비스 제공자의 인증 서버에 의한 입증 내역의 기록은 ALChain의 블록 생성 주기(예: 3초)만큼 최장 지연될 수 있지만 같은 사용자의 연속된 입증 요청이 이렇게 짧은 주기로 반복되는 것은 사실상 불가능하기에 연속 요청으로 인한 사용자 편의성 저하는 우려할 이유가 없다.

서버는 프로토콜 2의 단계 4만으로 사용자 제공 내용의 입증 여부를 결정할 수 있다. 단계 4의 세부 단계 중 암호화 연산 포함은 4.2, 4.3, 4.4이며, 각 세부 단계의 연산 수는 표 1과 같다.

표 1. 단계 4(프로토콜 2) 세부 단계의 암호화 연산 횟수

단계	해시(h_1, h_2) 연산	서명 검증 연산	비고
4.2	$t \times 2 + t \times 2$	0	t : 요소 수
4.3	1	0	
4.4	0	1	

한편, 비트코인 트랜잭션의 input 당 스크립트 실행 시 최소 1회(P2PKH의 공개키 또는 P2SH의 스크립트)의 SHA-256 해시 연산과 최소 1회의 ECDSA 서명 검증(P2PKH 또는 P2SH의 UTXO 소유자 입증용)이 필요하다. 그러므로 t개 input의 비트코인 트랜잭션 대비하여, 단계 4가 처리 속도가 빠른 해시 연산 외에는 암호화 연산 부담이 더 크지 않다고 할 수 있다. 비트코인 성능 연구로, 범용 PC(Core i7 8700 3.2GHz, 6 core CPU)로도 Bitcoin Core 23.0 소프트웨어 기준, 초당 평균 30,359 트랜잭션 검증이 가능함이 알려졌다[12,13]. 따라서 서버급 컴퓨터라면 30,000건/초 이상 수준(참고: Visa 카드 평균

1,700건/초, 최대 65,000건/초 처리)으로 제안 기법의 입증 처리를 충분히 달성하리라 판단된다.

VII. 결 론

빈번한 서버 해킹 사태에 서버 자체에 대한 보안 수준을 높이는 것이 유일한 해결책일지에 근원적 의문이 제기되고 있다. 제안 기법을 쓰면 서비스 제공자는 민감한 개인정보의 서버 보유 없이도 사용자가 능동적으로 제시한 정보가 올바르고 유효함을 블록체인을 통해 검증할 수 있다. 그 결과로 사용자는 사이트별 회원 가입의 필요가 없어지며 서비스 제공자 역시 고객 정보를 안전하게 저장 관리할 부담이 없어진다. 서버가 서비스 제공을 위해 꼭 보유해야 하는 정보가 있는 업종을 제외한 대다수 서비스 영역에서 제안 기법은 광범위하게 사용될 수 있다. 예로, 온라인 쇼핑에서 구매자의 결제와 배송지 정보, 여론조사 응답자의 거주지와 성별 및 연령대 정보, 렌터카 이용자의 운전면허 및 보험 관련 정보, 교사 지원자의 성범죄 전과 없음 등과 같이 인증이나 자격 검증이 필요한 상황을 들 수 있다.

REFERENCES

[1] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” https://bitcoin.org/bitcoin.pdf (accessed Oct. 02, 2025.)

[2] L. Yu, M. He, H. Liang, L. Xiong, and Y. Liu, “A Blockchain-Based Authentication and Authorization Scheme for Distributed Mobile Cloud Computing Services,” *Sensors*, 23(3), 2023.

[3] A. Garba *et al.*, “LightCert4IoTs : Blockchain -Based Lightweight Certificates Authentication for IoT Applications,” *IEEE Access*, vol. 11, pp. 28370-28383, 2023.

[4] C. McCabe, A. Mohideen, and R. Singh, “A Blockchain-Based Authentication Mechanism for Enhanced Security,” *Sensors*, 24(17), 2024.

[5] N. Singh and A.K. Das, “TFAS: Two Factor Authentication Scheme for Blockchain enabled IoMT using PUF and Fuzzy Extractor,” *J. of Supercomputing*, vol. 80, pp. 865 - 914, 2024.

[6] 박준철, “개인정보의 위임 제공 및 데이터 기밀성을 보장하는 블록체인에 제공 정보의 저장,” *스마트미디어저널*, 제11권, 제10호, 76-88쪽, 2022년 11월.

[7] P. Soni, S. Islam, A. K. Pal, N. Mishra, and D. Samanta, “Blockchain-based User Authentication and Data-Sharing Framework for Healthcare Industries,” *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 4, pp. 3623-3638, July-Aug. 2024.

[8] L. Yang, R. Jiang, X Pu, C. Wang, Y. Yang, M. Wang, L. Zhang, and F. Tian, “An Access Control Model based on Blockchain Master-Sidechain Collaboration,” *Cluster Computing*, vol. 27, pp. 477-497, 2024.

[9] A.M. Antonopoulos and D.A. Harding, *Mastering Bitcoin: Programming the Open Blockchain*, 3rd ed., O’Reilly Media, 2023.

[10] NIST Computer Security Resource Center, Hash Functions, SHA-3 Project, https://csrc.nist.gov/Projects/hash-functions/sha-3-project (accessed Sept. 29, 2025.)

[11] E. Andreeva, B. Mennink, B. Preneel, and M. Skrobot, “Security Analysis and Comparison of the SHA-3 Finalists BLAKE, Grostl, JH, Keccak, and Skein,” *Progress in Cryptology - Africacrypt*, LNCS vol. 7374, 2012.

[12] How many bitcoin transactions can be verified per second on commodity hardware in 2020?, https://bitcoin.stackexchange.com/questions/95339/how-many-bitcoin-transactions-can-be-verified-per-second-on-commodity-hardware-i (accessed Nov. 07, 2025.)

[13] J. Lopp, “Bitcoin Core Performance Evolution,” https://blog.lopp.net/bitcoin-core-performance-evolution (accessed Nov. 07, 2025.)

저 자 소 개

박준철(종신회원)
1986년 서울대 계산통계학과 학사
1988년 KAIST 전산학과 석사
1998년 U. of Maryland, College Park 전산학 박사
2001년 9월 ~ 현재 홍익대학교 컴퓨터공학과 교수
<주관심분야 : 블록체인 응용, 네트워크 보안>