IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

VAIM: Verifiable Anonymous Identity Management for Human-centric Security and Privacy in the Internet of Things

GYEONGJIN RA[1], TAEHOON KIM[2], AND IMYEONG LEE[3]

[1]Department of Software Convergence, Soonchunhyang University, Asan, 31538, South Korea (e-mail: rababi@sch.ac.kr)
[2]Department of Software Convergence, Soonchunhyang University, Asan, 31538, South Korea (e-mail: 20134101@sch.ac.kr)
[3]Department of Software Convergence, Soonchunhyang University, Asan, 31538, South Korea

Corresponding author: Imyeong Lee (e-mail: Imylee@sch.ac.kr).

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the High-Potential Individuals Global Training Program(2020-0-01596) supervised by the IITP(Institute for Information communications Technology Planning Evaluation) and the BK21 FOUR (Fostering Outstanding Universities for Research)(No. :5199990914048) and the Soonchunhyang University Research Fund.

ABSTRACT The human internet of things (HIoT) is a promising trend that adopts a user-centered vision to improve life quality by interacting with heterogeneous physical and virtual entities and the internet. However, It refers to exchanging contextual data between collaborative entities that raise privacy concerns. Emerging blockchain technology allows a digital identity management system (IDM) to be deployed in it, which largely alleviates the problems caused by the centralized third party. Still, its inherent transparency and lack of privacy pose a considerable challenge to IDM. We propose verifiable anonymous identity management (VAIM) connecting privacy channels between users by constructing identity verification and access control provisioning via user-centric decisions and an anonymous identity management system. This work has the following contributions: (1) We establish a novel IDM system by analyzing the existing scheme. In this regard, we improve the traditional claim identity model in blockchain by implementing zero-knowledge proof (ZKP) algorithms to achieve identity unlinkability, essentially preventing the disclosure of attribute ownership. (2) We implement a system that includes blind ordered multi-signature (BOMS) protocol, which allows users to processes efficiently and trusts the verification of anonymous transactions. (3) Finally, specific ZKP-based algorithm (commonly used practical ZKP such as Camenisch and Lysyanskaya signature (CL-Signature) and zero-knowledge succinct non-interactive argument of knowledge (ZK-SANRKS)) implementation and various environment performance evaluation and security analysis show that our scheme achieves efficient privacy protection and a broader application scope compared with the prior model. To the best of our knowledge, existing zero-knowledge proof-based IDM has not developed or compared each scheme before.

INDEX TERMS Anonymous identity management, Human-centric internet of things, membership service provider, privacy, verifiable credential.

I. INTRODUCTION

The internet of things (IoT) is dominated by massive amounts of content-oriented traffic, intensive interactions between numerous people, and heterogeneous communication between hosts and smart entities [1]. It requires millions of services with stringent real-time requirements and flexibility to connect everyone and everything. Nevertheless, consumption of IoT products and services remains above expectations [2]. Users are the essential entities in IoT systems because they are both data sources and consumers. Therefore, the human-centric internet of things (HIoT) is a promising trend that adopts a user-centered vision to improve life quality by interacting with heterogeneous physical and virtual entities and the internet. Miranda et al. [3] defined the internet of people (IoP) as easily integrated into the IoT and bringing the IoT closer to people to utilize its benefits fully. People as a service (PeaaS; hereafter, PS) is a mobile-centric computing model that can generate, maintain, and securely provide users' social-logical profiles as a service to third parties. PS relies on smartphones to emphasize smartphone functionality

and infer and share sociological profiles.

The advantage of this is that the continuous supply of data from the community provides the big data team[not understood; please clarify] [4] and increases the amount and diversity of contextual data, improving service intelligence and adaptability to the contextual needs of users [4]. However, according to a previous survey [5], 43% of users said they were afraid to use their data. In addition, 18% found that the connected entities did not work, and 8% felt unreliable. A second survey [6] indicated that 33% of users have concerns regarding what is being done with the data collected by IoT entities, 19% believe that these entities quickly become outdated, and 17% think that they are not very efficient or reliable. However, the medical ecosystem is becoming global, and demand is increasing rapidly. Therefore, the two main issues of trust and privacy are significant. Privacy concerns are related to the protection of users' data. The vast amounts of data collected by sensors and connected entities are typically stored and exposed in the cloud. It refers to the exchange of contextual data between collaborative things that raise privacy concerns. Users should be able to control and choose whether to grant access to their information. The security solution manages through an intelligent identity access management (IAM) mechanism that includes user identification and data attribute elements. These systems aim to achieve privacy through user-centered decisions, control access to information generated from user devices, and provide awareness of the use of data provided to third parties. At the same time, data that can be identified through anonymization technology are hidden. We can consider the following IAM implementation from the perspective of the awareness of HIoT discussed previously [3].

Considerations:

- The need for a human-centric perspective on digital consenting: Many people who engage in online activities are aware that their personal data are being collected and shared, but there is no evidence that this will lead to people's willingness to provide their personal data [3]. Users need to improve their privacy awareness by controlling their data and recognizing the flow of data usage.
- Fairness matters-human-centric perspective and marginalized people: End-user authorization from a human-centered perspective should be considered a universal approach in the design, implementation, evaluation, and release of consent acquisition mechanisms for everyone. Data need to provide and process through the interaction between the device and an external cloud server, convenient and with minimal user intervention. Mutual authentication and anonymity should be enforced using cryptographic tools.
- Enacting consent-consenting as a sociocognitive action: A human-centered approach is needed to obtain consent. To develop a basic human-centered framework for evaluating existing consent acquisition mechanisms, we apply blockchain's decentralization, autonomous consensus, and self-sovereignty.

We enable to find out various challenge works for mentioned issues and considerations. A computing device recognition (CDR) algorithm for an automated and secure user and device identification was proposed [7]–[9]. It allows the identification of users and access to various services offered by heterogeneous devices. At its core, the concept of privacy technology is that various user-centric decisions are made based on inputs received from the various sensing and monitoring devices that make up the human-centric sensor network (HCS-N). After that, due to single centralized errors of monitoring devices, collusion attacks in the network, and the absence of self-sovereignty, blockchain-based decentralized identity (DID) is in the spotlight as a distributed Identity Management (IDM). However, there are still cryptographic factors to consider when constructing a blockchain-based privacy system.

- Identity privacy: Privacy concerns can arise if the blockchain contains personally identifiable information. A single administrator of the private blockchain can identify users and track or verify transactions. Besides, sharing the blockchain between different business organizations can generate serious identity privacy concerns. In the blockchain, identity behavior can inadvertently disclose or leak information to other organizations that perform transactions on the same blockchain [10].
- Self-sovereignty through access control: Access control in a blockchain with shared permissions (read access and write access) is critical to the value of the blockchain. We prioritize infrastructure and service security over the privacy of entities using the infrastructure. A new approach needs beyond the traditional enterprise access control system, which establishes that individuals control access to their information and manage proactive data to improve privacy. [11].
- Selective disclosure and transaction privacy: The blockchain's ledger poses a potential threat concerning inferring or re-identifying its identity [12]. However, it is inefficient to provide confidentiality throughout the message. Also, users need to disclose the identities associated with the transaction in question (e.g., for anti-money laundering (AML) compliance) while protecting the transaction's privacy in a non-connected form. Therefore, personal information is stored separately. Sub-certificates are issued through selective encryption, such that individuals transact through multiple real identities and non-connectable transaction certificates on the blockchain. Individuals give identities and unconnectable transactions on the blockchain but claim to be the correct user and can later disclose their identities on the blockchain, which are granted the same sharing rights (e.g., in the case of legal issues) [13].

This paper is the ordered multi-signature [14] and the Identity Mixer system based on the blockchain DID and hyperledger indy system [15]–[17]. Also, It is based on the

zero-knowledge succinct non-interactive argument of knowledge (ZK-SNARKs) and circuit and quadratic arithmetic program (QAP) polynomial scheme [18]. These were discussed previously in-depth [19]. This paper proposes the verifiable anonymous identity access management (VAIM) system that considering accuracy and privacy trade-off. Here, we apply and extend previous work [15] dealing with identity and privacy in the blockchain. The paper is structured as follows. In Section 2, we research related work state-of-arts identity and privacy in the blockchain. In Section 3, we describe the system security requirements and the underlying technology for VAIM. In Section 4, we explain the VAIM architecture and protocol. The design of VAIM makes it possible to construct a permissioned blockchain with identity and user management through IAM on various blockchains to protect user privacy independently of the underlying blockchain system. In Section 5, we develop the VAIM execution environment through hyperledger fabric (HLF). In Section 6, We analyze our system for security and efficiency through comparisons of this work with traditional schemes based on Sections 2 and 3 requirements. Finally, we present a summary of our results and future goals in section 7.

II. RELATED WORK

We look at the related works of IDM, which is the essential secure gateway to manage data in HIoT. We find out ZK-SNAKRs and Camenisch, and Lysyanskaya signature (CL-Signature), which are commonly used in zero-knowledge proof (ZKP) based IDM. It supports our insights into human-centric IDM security and system design goals.

A. IDENTITY MANAGEMENT

Generally, IDM is applied anonymous technology to protect user attributes because of identifies and manages users. We look through various techniques Table 1 classified according to the anonymity level to ensure privacy in IDM.

1) Anonymity Level 0 3-based IDM

The existing public key infrastructure (PKI) system binds the public key and the user's identification information to use the public key [20]. Therefore, it does not provide anonymity. Bitcoin [21] is a pseudonym system that uses the hashed value of the public key as an address and provides anonymity. However, if the same address is used repeatedly, the user's behavior can be tracked. Therefore, the identifier's connection is terminated using multiple certificates or various same addresses. However, the user needs to use various keys for authentication. Group signature [22] cancels a single attribute by using a single group key generated by the manager. Still, there is a problem with critical group updates and the group manager's overhead.

2) Anonymity Level 4 5 based IDM

The anonymous credential system (ACS) [23] securely stores registration and private keys through a trusted third party (TTP) or trusted platform module (TPM) and terminates the user's connectivity by deriving sub-attributes. Specifically, direct anonymous attestation (DAA) [24] is a remote anonymous attestation protocol based on group signing and zero-knowledge attestation. It is based on three entities and two different steps. In DAA and ACS, TTP must provide high availability while securing all transactions. Also, collusion between the certification authority (CA) and the verifier may violate privacy. Shin et al. proposed a modified remote anonymous authentication protocol for use in mobile environments [25]. Anonymous authentication was proposed using the CL-Signature. Unlike the group signature scheme, DAA does not provide a signature open operation for privacy protection [26]. Thomas et al. proposed a DAA-based enhanced privacy identity (EPID) scheme [15]. Therefore, using CL-Signature ZKP and group signature is the same, but Thomas et al. used the value of CL-Signature as group public key to issue private keys to group members. It efficiently executes multiple issuance and verification of user's private keys through a group public key. ChainAnchor provides the ability to disconnect transactions belonging to entities in the blockchain. In the public blockchain, it is possible to provide the same functions as the permissioned blockchain in a hybrid form. However, in CL-Signature-based DAA, the verifier must calculate the proof's verification through the pairing operation. It has computational overhead when the user verifies the proof after receiving the property from the upper TTP in the form of proof. To this end, Z-Cash [27], Hawk [28], [29] use ZKPs based on blockchain and arithmetic circuits so that verifiers can perform verification very efficiently. ING Bank described how to implement zero-knowledge range proof (ZKRP) protocol in Ethereum [30]. Therefore, ZKRP can be applied to many types of decentralized applications with numeric intervals, and other requirements, such as e-voting systems and e-auction systems [19], [31].

TABLE 1. Comparison of anonymity level for privacy in authentication

Anonymity level	Anonymous authentication techniques
0	PKI-digital signature based authentication
1	Pseudonym based anonymous authentication
2	Multilevel authentication
3	Group signature based authentication
4	DAA, ACS
5	Zero knowledge certificate, certificate based ring signature

B. ZK-SNARKS & CL-SIGNATURE

ZK-SNARKs and CL-Signature are a representative scheme that provides strong anonymity in IDM. ZK-SNARKs generate concise proofs that verifiers can efficiently verify proofs. Unlike standard non-interactive zero-knowledge proof (NIZK), SNARK guarantees knowledge soundness, a powerful concept compared to standard soundness. The knowledge soundness of ZK-SNARKs is not a black box and is achieved under the assumption of knowledge. The most efficient ZK-SNARKs to date have been proposed by Groth [32] at Eurocrypt 2016.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

TABLE 2. ZK-SNARKs and CL-Signature performance comparison

scheme	Proof/sig gerneration cost	Proof/sig gerneration time (s)	Verification cost	Verification time (s)
Groth 16	$4E$	0.05468	$3P + E$	0.14
Camenish, Lysyan-skay	$2E$	0.02734	$4P + E$	0.18211

E : exponentiations (0.01367 s), P : Pairing (0.04211 s)

On the other hand, CL-Signature can prove that the prover integrity provides the calculated value required by the verifier without revealing a message. CL-Signature was proposed in 2002 by Camenisch, and Lysyanskaya [33]. This product also works in bilinear groups. The proof of CL-Signature consists of 1 element of G_1 and one element of G_2, and the verifier must confirm one equation where three pairs dominate.

The following Table 2 shows the estimated time required to execute the theoretical calculation of ZK-SNARKs and CL-Signature [34]. Unlike the isomorphic calculation of CL-Signature, ZK-SNAKRs has the advantage of concise verification by performing simple arithmetic circuit operations. It provides efficiency in a peer-to-peer environment such as a blockchain where multiple verification nodes exist.

In this paper, we research, implement, and analyze ZK-SNARKs and CL-Signature-based IDM frameworks [32], [33].

III. PRELIMINARY

A. BLIND ORDERED MULTI SIGNATURE

Sequential signatures provide a multi-party digital signature scheme that allows multiple signers to sequentially generates compact fixed-length signatures, which allows signers to prove a standard message and the order in which they were signed [14]. Sequential signatures provide a multi-party digital signature scheme that allows multiple signers to sequentially generates compact fixed-length signatures, which allows signers to prove a standard message and the order in which they were signed [14]. We introduce blind ordered multi-signature (BOMS) among sequential signatures. The user inputs the signature signed by the previous user and outputs an ordered signature. BOMS sets with six tuples as follows [14]:

- OPg: A parameter generation algorithm that returns global information i, which can be run by a TTP or standards body.
- OKg: A key pair generation algorithm that generates a public key-private key pair (pk, sk) through global parameters entered.
- BM: The user selects random integers w and r and calculates the secret key by exponentially multiplying it.
- $OSign$: This is the signature algorithm that the user executes when entering the message $m \in \{0,1\}^*$, OMS through the secret key sk. Verification is performed through the public key list $L = (pk_1, \ldots, pk_{i-1})$ and returns OMS 0, or $\perp$ if the input is invalid.
- UBM: The user calculates the OMS value of unblind BM using pk.
- OVf: Verification algorithm for entering public key list $(pk_1, \ldots, Pk_n)$. OVf list, message m and OMS σ' returns significant bits.

B. NON-INTERACTIVE ZERO-KNOWLEDGE PROOF

ZK-SNARKs can implement ZKP in a blockchain environment. In the case of a blockchain transaction using ZK-SANRKs, the transaction's validity can be notified to other nodes other than the sending/receiving node without exposing the recipient, sender, and transmission amount [25]. The verifier learns nothing except the validity of the computation. The proof is tiny compared to the calculation. The proofs are generated without interaction with the verifier and are publicly verifiable strings. Soundness is guaranteed only against a computationally bounded prover. The proof cannot be constructed without access to a witness [18].

- Homomorphic hidings (HH): A HH $E(x)$ of a number x is a function satisfying the following properties.
  1) For most values of x, given that it is hard to find x.
  2) Different inputs lead to different outputs, so if $x \neq y$, then $E(x) \neq E(y)$.
  3) If someone knows $E(x)$ and $E(y)$, they can generate the HH of arithmetic expressions in x and y. e.g., they can compute $E(x + y)$ from $E(x)$ and $E(y)$.
- Blind evaluation of polynomials: Blind evaluation can be performed using HH as follows.
  1) The prover sends hidings to Alice. $E(1)$, $E(s)$, $\ldots$, $E(s^d)$
  2) $E(P(s))$ is calculated using the values transmitted in the first step and sent to the prover. As E supports linear combinations, the verifier can do this easily. By sending values only hidings, the verifier cannot know about s, and Bob cannot know about P.
- Arithmetic circuits and reduction QAP: ZK-SNARKs cannot be directly applied to any computational problem. Instead, it must be transformed into the correct form so that this problem can be used. This form is called the QAP, and it is essential to change the function code into this form. It is another process that can be executed along with converting the code of a function to QAP.

For ZK-SNARKs, we adopt the definitions from [18].

Definition 1. A relation generator RG returns a polynomial time decidable relation $R \leftarrow RG(1^\lambda)$ given a security parameter λ. We claim ω is a witness to the statement (I/O) ϕ being in the connection for $(\phi, \omega) \in R$. The statement ϕ in $SAVER$ is made up of $\phi = M \cup \hat{\phi}$ for message statements, $\{m_1, \ldots, m_n\}$ by $M = (m_1 || \ldots || m_n)$ and $\hat{\phi} = \{\phi_{(n+1)}, \ldots, \phi_l\}$ for arbitrary statements, with l representing the number of statements.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

A ZK-SNARKs is made up of four tuple [35]: $Setup$, $Prove$, $Vefify$, and $SimProve$.

– $(crs, \tau) \leftarrow Setup(RL)$: returns a common reference string crs and a simulation trapdoor τ from a connection $R \leftarrow RG(1^{\lambda})$.

– $\pi \leftarrow Prove(crs, \phi, \omega)$: returns a proof pi using a standard reference string crs, a relation RL, a statement and witness in the relation$(\phi, \omega) \in R$ as inputs.

– $0/1 \leftarrow Veify(crs, \phi, \pi)$: takes three inputs: a standard reference string crs a proposition ϕ and a proof π and returns 0 (reject) or 1 (accept)

– $\pi \leftarrow SimProve(crs, \tau, \phi)$: returns a proof pi using a standard reference string crs, a simulation trapdoor τ, and a statement ϕ as inputs.

It meets the following properties: completeness, computation knowledge soundness, statistical zero-Knowledge, and succinctness [35]:

Properties 1. Completeness: A prover with a witness will persuade the verifier given a valid statement. For all $\lambda \in N$, for all RL and for all $(\phi, \omega) \in R$,

$$Pr[(crs, \tau) \leftarrow Setup(RL), \pi \leftarrow Prove(crs, \phi, \omega) : Verify(crs, \phi, \pi) = 1] = 1$$

Properties 2. Computation knowledge soundness: The prover can not make the verifier accept a wrong statement $(\phi, \omega) \notin R$ except with some small probability. The above definition requires the soundness error to be negligible in the security parameter, λ. By increasing k, the soundness error can be made arbitrarily small. If the soundness error is 0 for all λ, we speak of perfect soundness. The prover must know a witness and that this knowledge can be extracted from the prover effectively by a knowledge extractor. Proof of knowledge implies that for every malicious prover $\mathcal{A}$ that generates an approving proof, there must be an extractor χ_A that outputs a legitimate witness given the same input as $\mathcal{A}$. Formally, the argument scheme $\Pi snark$ is called computationally knowledge sound if there exists a probabilistic polynomial time (PPT) extractor χ_A for any PPT adversary $\mathcal{A}$, so that $\Pi snark, \mathcal{A}, \chi_A(\lambda)$ is negligible function v, Advsound $\Pi snark, A, \chi_A, (\lambda)$,

$$Pr[(crs, \tau) \leftarrow Setup(RL), (\phi^*, \pi^*) \leftarrow A(crs), \omega \leftarrow \chi_A(transA) : Verify(crs, \phi^*, \pi^*) = 1 \wedge (\phi*, \omega) \in R] = v(\lambda)$$

Properties 3. Statistical zero-knowledge: A non-interactive proof system $(Setup, Prove, Verify)$ is zero-knowledge, if there exist a simulator $Sim = (Sim_1, Sim_2)$, such that for all non-uniform polynomial time adversary $\mathcal{A}$,

$$Pr[(crsSetup(RL) : A^{Prove(crs,...)}(crs) = 1] \equiv Pr(crs, \tau)Sim_1 : A^{SimProve(crs,\tau,...)}(crs) = 1]$$

Properties 4. Succinct: There exists a fixed polynomial $p(\dots)$ independent of R such that for every large enough security parameter $\lambda \in N$, every time bound $T \in N$ and every instance $y = (M, x, t)$ such that $t <= T$.

IV. VIAM FRAMEWORK

We propose human-centric IDM frameworks based on blockchain and ZKP. First of all, blockchain such as ledger provides verifiable public trust. Also, BOMS provides ordered verification, for which each participant has accountability. Second of all, data access is controlled by the private key, ensuring self-sovereign privacy. Finally, It provides strong anonymity through ZKP-based certificates. Users can use this system to ensure their anonymity and use the web's services.

A. SYSTEM REQUIREMENTS

The system requires the following characteristics:

- Privacy: Users legitimately request the issuance of qualifications through the blockchain, present them and selectively disclose their information through ZKP.
- Valid public verification possibility: Users and verifiers can repeatedly verify the validity of information at any time through the blockchain.
- Zero-knowledge: The verifier cannot know the user's information other than the validity of the arguments of ZKP.
- Anonymity: Through the anonymous entitlement system, a user generates a subset of attributes to revoke user attributes connected offline.
- Conditional disclosure and traceability: When a problem occurs, the user reveals their own identity to prove their integrity, or the certificate issuer and verifier track malicious users to improve the internal reliability of the system.
- Based on the non-polynomial problem: Strong Rivest, Shamir, and Adleman (RSA) assumption [11] and decisional Diffie-Hellman assumption, so it is secure in random oracle model.

B. SYSTEM MODEL

1) Entities

- User: As an information provider, this is a subject that requests the issuance of credentials.
- Verification server: Determines the network execution qualification by checking the validity of the information of the entities participating in the blockchain.
- System management: As the system administrator inside the blockchain, all subjects generate a key pair and issue certificates for use in the blockchain network.
- Entry server: Performs an interactive network of internal operations of the subject's anonymous credential verification protocol and external result values.
- root CA: Stores user attributes and distributes public key pairs and user attribute certificates. Tokens are generated and distributed to the issuing server and users.
- Intermediate CA (I-CA): Distributes a public key pair and anonymous transaction certificate. It generates the verification key and the verification key of the verifier through which the user's zero-knowledge verification and anonymous certification are generated.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

TABLE 3. System parameters

Symbol	Quantity
PK_*	Public key of *
SK_*	Private key of *
PRK_*	Key for generation proof of *
VK_*	Key for verification proof of *
w_*	Credentials of *, secret input data to generate proof
crs	Common reference string, common attributes of keys shared by the prover and verifier
$0, 1^l$	All sets of binary strings of length l
g'	Random generator for QR_n of n groups of quadratic redundant modular
g, h, s, Z, R_0, R_1	6 random integers to be chosen
r, ρ, γ	A prime number that satisfies the security parameters
π_*	Proof generated by *
τ	Trap door
ϕ	Inputs shared with the prover and verifier, definition of the statement to be proved
R	Relation

- Verifier: Verifies the user's anonymous certificate and proof.

2) System Parameters

The system parameters used in this system are shown in Table 3.

3) Cryptography Algorithm

The algorithms (length) used in this system are as follows.

- $l_{RSA}(1024)$: RSA key pair (length) generated by the registration server.
- $l_{DSA}(1024)$: Digital signature algorithm (DSA) signature (length) generated by the registration server.
- $l_{ECDSA}(1024)$: Elliptic curve digital signature algorithm (ECDSA) signature generated by the issuing server (length).
- $l_{ECC}(160)$: Elliptic curve cryptography (ECC) key pair (length) generated by the issuing server.
- $l_H(256)$: Hash function (secure hash algorithm-2 (SHA-2)) result (length).
- $l_T(16)$: Random number token (length) generated by the registration server.
- $l_c(162)$: ZKP required in the preparation process random prime number c (length).
- $l_r(1264)$,$l_\omega(1185)$: Random prime number (length) for private key generation.
- $l_v(1185)$: ZK-SNARKs generated by the issuing server.

4) System Architecture

The VAIM system architecture consists of five phases as follows.

- Offline enrollment: Each entity registers their personal information with the root CA to obtain a public key pair and certificate.
- System setup: Each entity generates a public key pair for communication and signing. At this time, it assumes that the registration server generates a public key and generates a parameter for ZK-SNARKs, delivers it to the object, and then destroys it.
- User registration: The prover requests and verifies the credential value. The smart device then verifies proof with the verification key.
- Issuing anonymous certificate: The entity requests the issuing server to issue an anonymous certificate. The issuing server then issues an anonymous certificate, signs it, and sends it to the entity. Moreover, it stores the fact that it was issued on the blockchain.
- Anonymous certificate verification: The verifier verifies the validity of the proof values and anonymous certificate registered in the blockchain and accepts the final prover's qualification.

C. PROTOCOL

In this section, we describe the detailed protocol operation of VAIM. It includes the following five phases.

1) Offline Enrollment

In the offline enrollment phase, the root CA receives an ID and password (PW) from all entities. The root CA uses RSA encryption to generate a key pair for each entity. After that, a certificate is generated using the private key of root CA, the public key of the entity, and the digest. The root CA sends the key pair and certificate to all entities. All entities receive a key pair and certificate from the blockchain root CA and are prepared to access the blockchain network. (See algorithm 1)

Algorithm 1 Offline Enrollment

Input: $id, password$
Output: $PK_*, SK_*, certificate_*$

```
if Verifyid(id) = False then
    pk_*, sk_* ← KeyGen(KeyGenOpts)
    certicate_* ← Sign(sk_RootCA, pk_*, digest)
    return pk_*, sk_*, certificate_*
else
    break
end if
```

2) System Setup

In the system setup phase, the system manager takes a security parameter, k, integer n, as input. It chooses a bilinear map system, PG. Each system entity makes a public key pair through the $publicparameter, PG, h$. (See Algorithm 2)

3) User Registration

In the user registration phase, the registration server receives the ID and PW from the user's decentralized application (DApp) and verifies that it is a registered user. If registered users, the enrollment server uses the ECC algorithm to generates a key pair and certificate. The user receives the generated key pair and certificate and is prepared to access the blockchain network. After that, the registration server generates the necessary user token to request the user's

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

Algorithm 2 System Setup

Input: n, p, g, h, k
Output: OPg, OKg, PK, SK
compute$\xrightarrow{\$} PG = (p, G_1, G_2, e)$
select $\xrightarrow{\$} h_1 : \{0,1\}^* \rightarrow G_1, h_2 : \{0,1\}^* \rightarrow Z_p^*$
compute$\xrightarrow{\$} pp = \{G_1, G_2, e, p, g, h_1, h_2\}$
$U_d \in \cup(d \in [1, n])$
$h \in G,\ a, x \in Z_p$
compute $\xrightarrow{\$} g^a,\ h_i = h^{a_i}$
for $i = 1, ..., n$ **do**
  $k_u = (g^a, h_1, h_2, \ldots, h^n), SK_u = a$
  $\alpha \in_R G_1, a_s \in_R Z_p^*$
  compute $\beta = g^(a_s)$
  $PK_s = (\alpha, \beta), SK_s = a_s$
end for
for $i = 1, ..., n$ **do**
  $I \xleftarrow{\$} OPg$
  $(PK_1, SK_1), \ldots, (PK_n, SK_n) \xleftarrow{\$} OKg(I)$
end for
return σ_0,L_0

Algorithm 3 User Registration

Input: $id, password$
Output: $PK_*, SK_*, E_{PK_*}(token)$
if $Verifyid(id) = False$ **then**
  $pk_*, sk_* \leftarrow KeyGen(KeyGenOpts)$
  return pk_*, sk_*
else
  $break$
end if
$token \leftarrow Rand(RandomNum)$
$E_{PK_{user}}(token) \leftarrow Encrypted(PK_{user}, token)$
$E_{PK_{issue}}(token) \leftarrow Encrypted(PK_{issue}, token)$
Transmit a token to the user and issuing server
return $E_{PK_*}(token)$

anonymous transaction certificate. The registration server encrypts the token using the user's public key and the issuing server. The encrypted token is sent to the user and the issuing server, respectively. (See algorithm 3)

4) Issuing Anonymous Certificate

In the certificate issuance phase, the user sends a token to the issuing server and authenticates the user. Suppose the registered user requests issuance of a certificate and receives a zero-knowledge certificate and an anonymous certificate generated by issuing server. The user sends a token to the issuing server. Then user sends a public parameter to generate the attestation. The issuing server generates a verification key VK in the CRS of the registration server. The issuing server generates a value containing the user's witness w or message m and sends it to the user. (See algorithm 4)

Algorithm 4 Anonymous Certificate Generation using ZK-SNARKs

Input: $token, privateinput, OPg, OKg, BM$
Output: $anonymous\ certificate, OSign$
if $Verifytoken(token)$ **then**
  $circuit \leftarrow GenerateTrustedSetup(circuit, alphas, betas, gammas)$
  $anonymous\ certificate \leftarrow GenerateACert(circuit, vk, proof)$
  Issue a transaction on the generation of an anonymous certificate
  for $i = 1, ..., n$ **do**
    $\sigma_i \xleftarrow{\$} Osign(sk_i, m, \sigma_{i-1}, L_{i-1})$
    $L_i \leftarrow (pk_1, \ldots, pk_i)$
    $Osign(sk, m, \sigma, L) \Rightarrow \perp$
  end for
  return $anonymous\ certificate$
else
  $break$
end if

5) Anonymous Certificate Verification

The user presents zero-knowledge proof and an anonymous certificate in the anonymous certificate verification phase, which the verifier verifies through the verification key. After that, the user transmits the proof value received from the issuing server to the verifier. Also, the verifier verifies using proof value proof. If verification is successful, the verifier returns True; otherwise, it returns False. (See Algorithm 5)

Algorithm 5 Anonymous Certificate Verification using ZK-SNARKs

Input: $anonymous\ certificate$
Output: $True/False$
if $Verifyproof(vk, proof, publicSignals)$ **then**
  return $True$
else
  return $False$
end if

V. IMPLEMENTATION

This section describes and evaluates the implementation developed in this paper. For this purpose, an anonymous certificate-based authentication system suitable for the permissioned blockchain is applied, and data are separated and stored, and processed to improve efficiency. In the method proposed here, a blockchain with guaranteed privacy is possible. The permissioned blockchain protocol using the NIZK is derived, and various scenarios (number of participants and verifiers) are derived.

A. IMPLEMENTATION FLOWCHART

As shown in Figure 1, the entity configuration is divided into user, registration server, issuing server, and service provider.

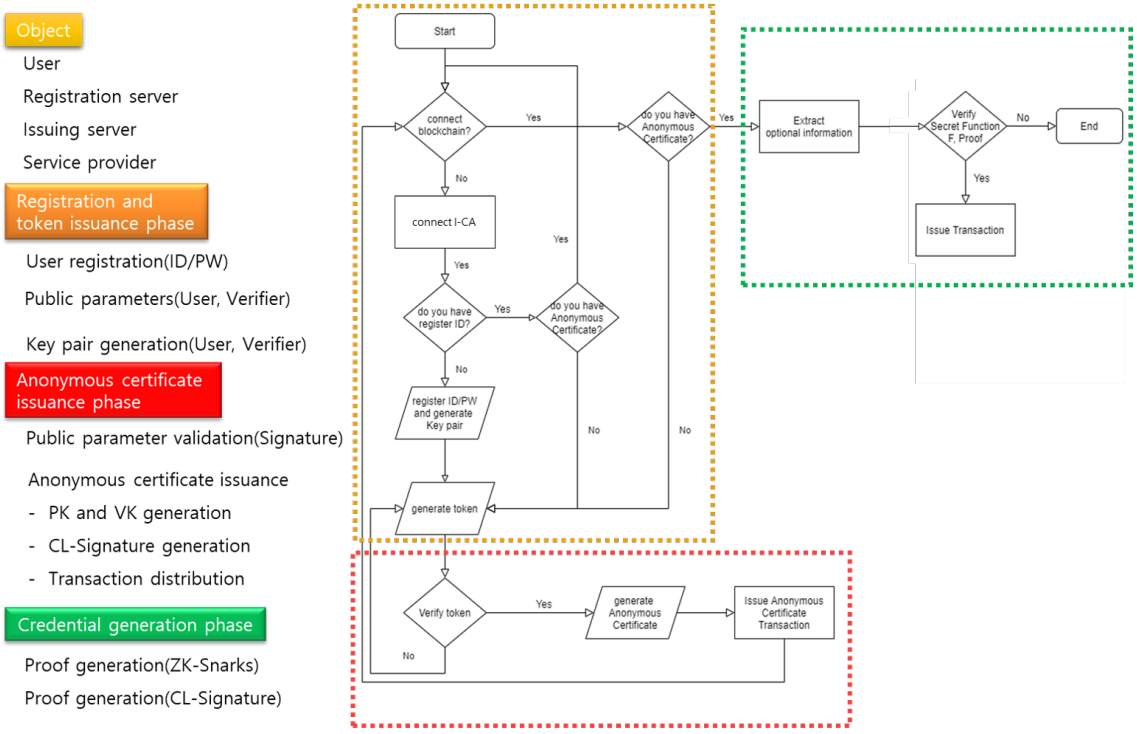

FIGURE 1. Implementation flowchart

The operation process is divided into three phases that are registration and token issuance, anonymous certificate issuance, and credential generation.

The registration and token issuance phase determine whether the user can access the blockchain network. If it is not possible to connect to the blockchain network, it registers an ID from the registration server. When ID and PW are registered, a public key pair is generated and transmitted. The user then requests the registration server to issue a token, and the registration server generates a token and publishes it to the user and issuing server.

In the anonymous certificate issuance phase, the user sends a token to the issuing server to request an anonymous certificate. The issuing server then verifies the token, and if it is confirmed that the registration server generates it, it generates an anonymous certificate using the NIZK. It is then issued to the user.

Finally, in the credential generation phase, the user participates in the blockchain network using the issuing server's anonymous certificate. After joining the blockchain network, the user sends an anonymous certificate to the service provider. Through NIZK, the user requests a legitimate service from the service provider. The service provider provides the service upon completion of verification.

B. FUNCTIONS USED IN IMPLEMENTATION

The implementation process is detailed into a total of 12 phases. The function used in the step is defined as Table 4.

TABLE 4. Functions used in implementation

Phase	Sender	Receiver	Function
1	User	Register	Request(ID string, PW string)
2	Register	Register	KeyGen(opts bccsp.KeyGenOpts)
3	Register	User	SendKeypair(pk string, sk string)
4	Register	Register	Rand(RandomNum string)
5	Register	User	SendToken(Token string)
6	User	Issuer	SendToken(Token string)
7	Issuer	Register	VerifyToken(Token string)
8	Issuer	Issuer	GenerateTrustedSetup(circuit circuitcompiler.Circuit, alphas [][]*big.Int, betas [][]*big.Int, gammas [][]*big.Int) GenerateACert(circuit circuitcompiler.Circuit, vk Vk, proof Proof) GenerateCLsig(message string, pkinfo PkInfo) GenerateACertCLsig(pkinfo PkInfo, clinfo CLInfo)
9	Issuer	User	SendACert(newACert json)
10	Issuer	All participants	IssueACert(newACert json)
11	User	Service provider	GenerateProof(Credential string)
12	Service provider	Service provider	VerifyProof(vk Vk, proof Proof, publicSignals []*big.Int) VerifyCLsig(X BN254, Y BN254, Z BN254, a BN254, A BN254, b BN254, B BN254, c BN254)

C. PROPOSED SCHEME IMPLEMENTATION

The system implementation proposed in this paper is described step by step here. An applied anonymous certificate-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

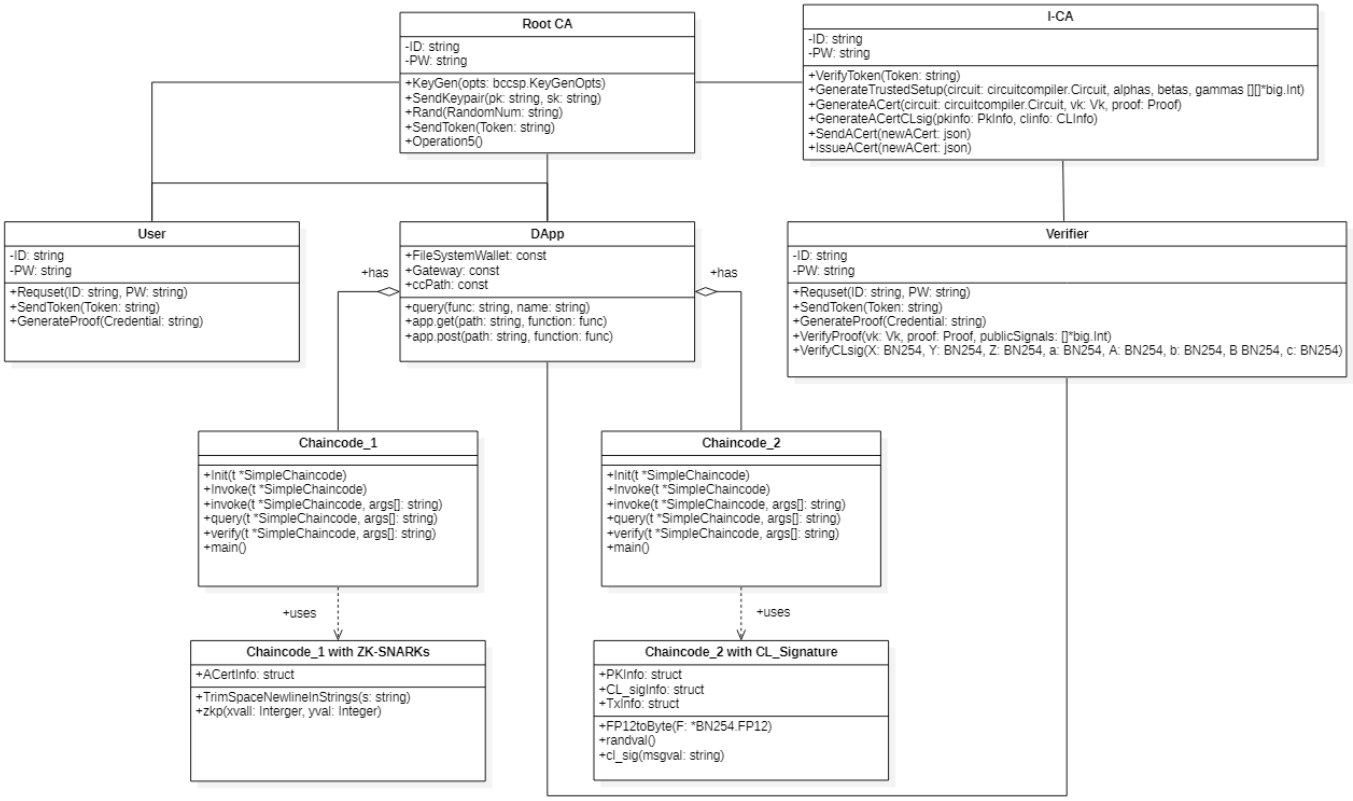

FIGURE 2. Class diagram of implementation

based credential system suitable for permissioned blockchain was developed and applied using development frameworks, such as HLF and Node.js.

1) Algorithm

Figure 2 shows a class diagram of the chaincode used for implementation. HLF implements ZK-SNARKs and CL-Signature chaincode. In the DApp class, various variables, such as the user's wallet, HLF connection, and chaincode path, are declared. Besides, the function receives chaincode function and parameters and executes chaincode from HLF. The results of the executed chaincode are output to DApp.

The Chaincode_1 with ZK-SNARKs class is a chaincode made to use ZK-SNARKs. For this, the components for anonymous certificates are declared in the ACertInfo structure, and it is implemented to operate ZK-SNARKs.

The Chaincode_1 class is a chaincode designed to run chaincode in HLF. The Init function initializes the chaincode. Also, the Invoke function determines and operates chaincode function. Also, invoke function instantiates chaincode. The query function allows the loading of data stored in CouchDB. The verify function verifies ZK-SNARKs.

The Chaincode_2 with CL-Signature class is a chaincode made to use CL-Signature. For this, the components for anonymous certificates are declared in the PKInfo, CL_sigInfo, and TxInfo structures. It is implemented such that CL-Signature can be operated.

The Chaincode_2 class executes the same as Chaincode_1. However, the verify function verifies CL-Signature.

2) root CA Administrator Registration

The highest root CA can manage I-CA by issuing I-CA certificates to I-CA. By registering the root CA administrator, the root CA administrator can manage the lower I-CA.

3) I-CA Administrator Registration

As an intermediate certification authority, I-CA is a CA that shares and manages the root CA tasks. Also, an administrator is registered who manages I-CA. The I-CA administrator

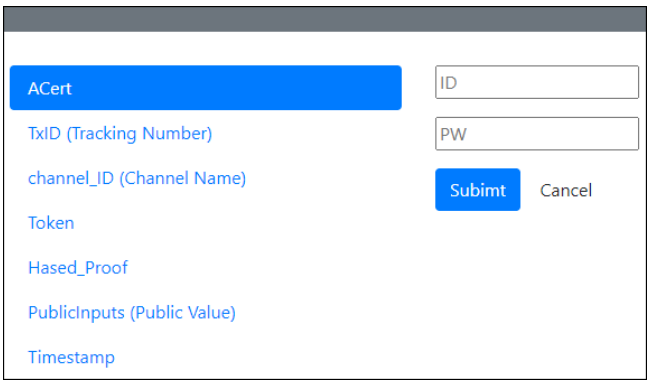

FIGURE 3. Login in DApp

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

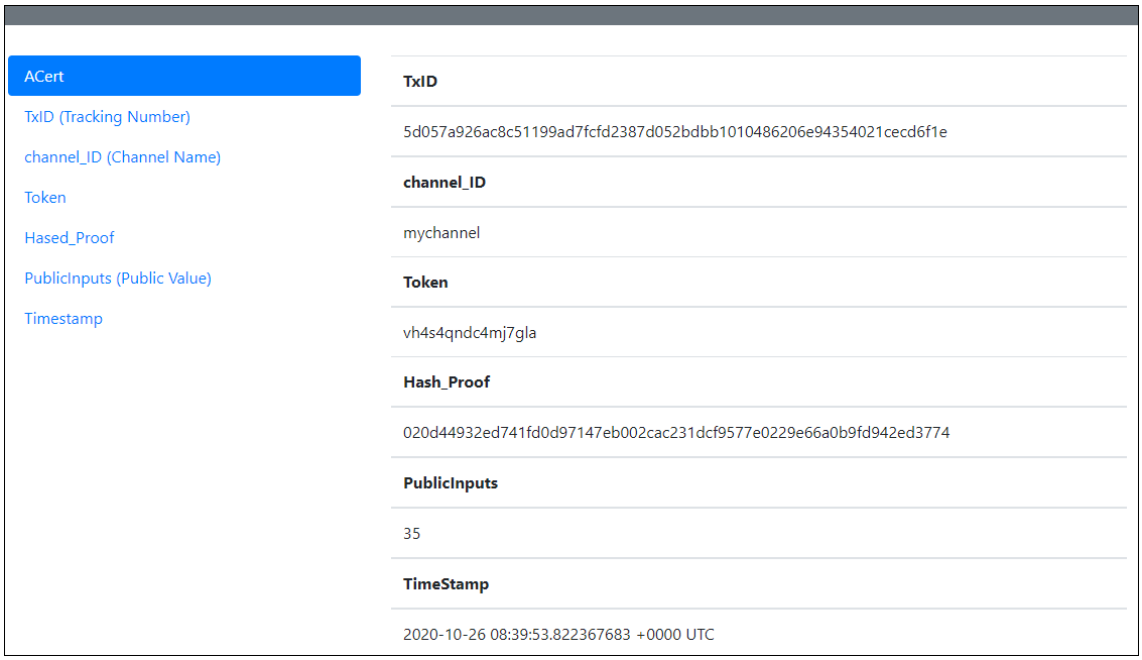

FIGURE 4. Component of an anonymous certificate using ZK-SNARKs

authenticates users who access the HLF network and issues certificates. I-CA has a registration server and an issuing server.

4) User's Key Pair, Certificate Generation, and Registration

For a user authenticated by I-CA, the registration server generates and transmits a public key pair and X.509. The contents of the generated key pair are stored and managed in the registration server database.

5) Token Generation and Transmission

The user requests the registration server to issue a token using the key pair and certificate from the registration server. The registration server verifies the user's key pair and certification and then issues and transmits a token. The user sends the token received from the registration server to the issuing server.

6) Zero-Knowledge Proof

Two ZKPs were implemented as chaincode to provide ZKPs and perform comparative analyses. The go-snark library was used to implement ZK-SNARKs and provided trusted setup, ZKP generation, and verification functions. In addition, the miracle library was used to implement CL-Signature and provided message signing, ZKP generation, and verification functions.

It was implemented as a DApp to quickly show the chaincode usage and data in the HLF network. After selecting an identity from the wallet, it is determined whether it is a registered user. Next, it connects to the HLF network, requests chaincode, and transmits the function and parameters to be used. The responded data are output to the DApp web page to be used and checked easily by users.

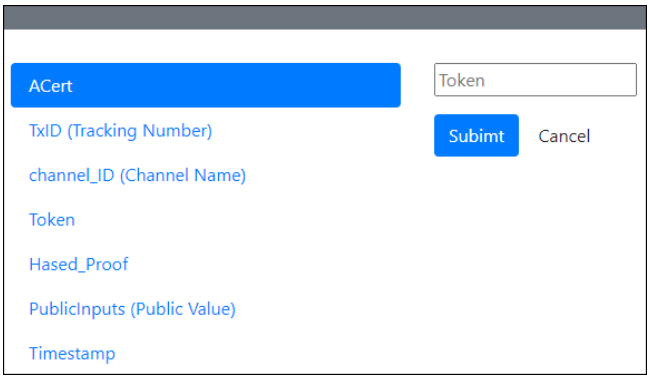

FIGURE 5. ZK-SNARKs verification

7) Implementation Result

We present the implementation results and the DApp results for ZK-SNARKs and CL-Signature. As shown in Figure 3, users can gain access by entering the registered user's ID and PW when accessing DApp.

Figure 4 shows a component of an anonymous certificate using ZK-SNARKs and contains the hashed proof value to provide the data and integrity required for later verification.

Figure 5 traces the transaction of the anonymous certificate by entering the transaction ID. The elements of the anonymous certificate are then checked to provide integrity and validation.

Figure 6 is a component of an anonymous certificate using CL-Signature. It contains a unique value to provide the data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

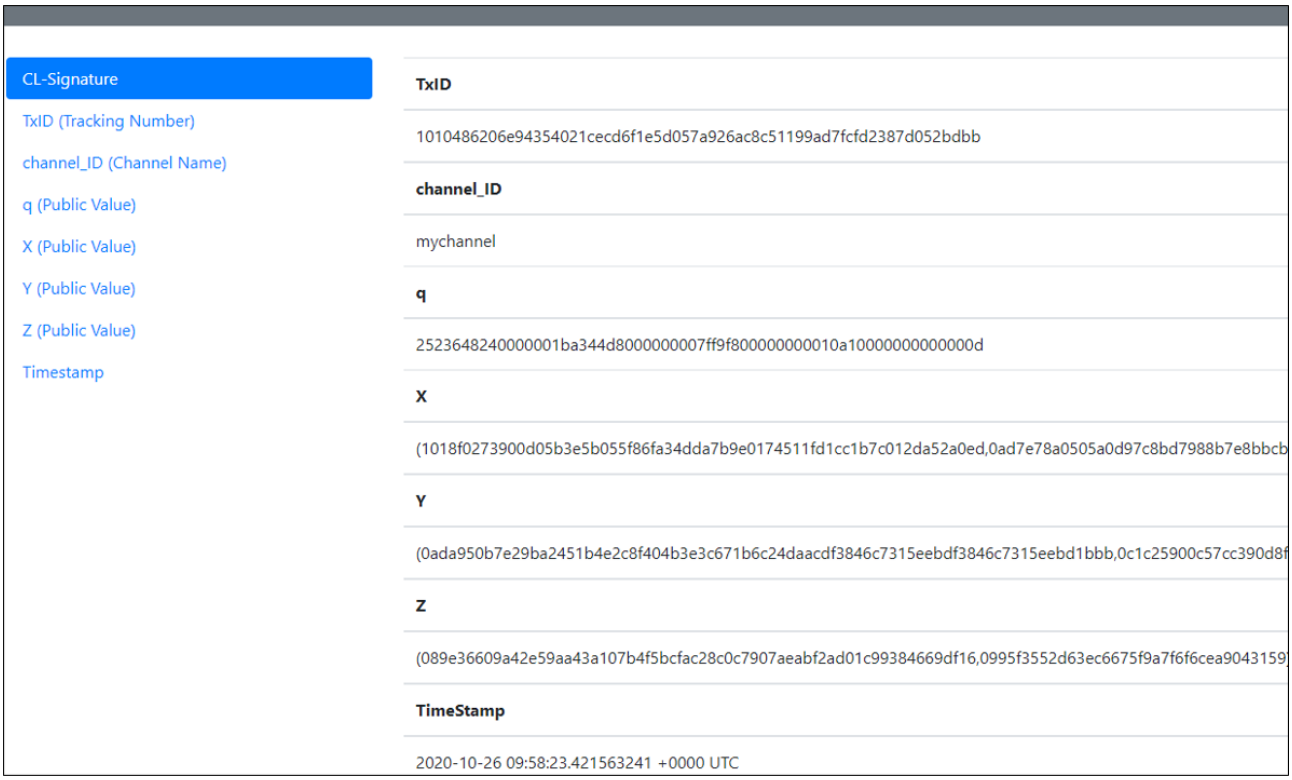

FIGURE 6. Component of an anonymous certificate using CL-Signature

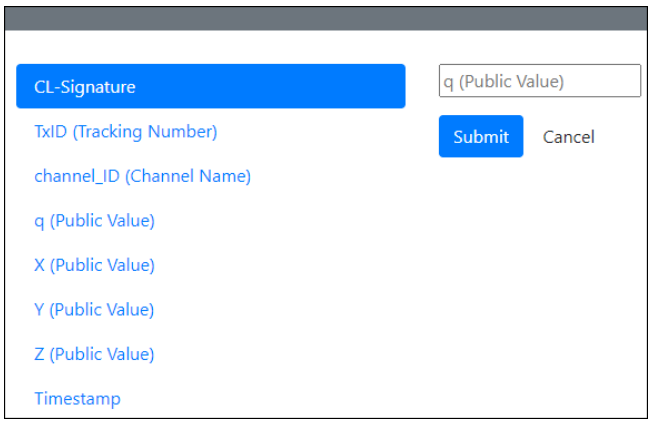

FIGURE 7. CL-Signature verification

and integrity required for later verification.

Figure 7 traces the anonymous certificate transaction by entering the transaction ID as shown in Figure 5. The elements of the anonymous certificate are then checked to provide integrity and validation.

D. IMPLEMENTATION SYSTEM, ENVIRONMENT CONFIGURATION

We describe the implementation system environment configuration. As shown in Table 5, the root CA, I-CA, and DApp used the cloud platform provided by Google. The user configured the implementation system using the basic specification by MacBook 2020. The verifier used a regular desktop personal computer (PC).

To implement a performance evaluation scheme for comparing ZKPs, we referred to hyperledger blockchain performance metrics, a hyperledger white paper, which defines basic terms and key indicators used to evaluate hyperledger's performance communicate results. It also defines guidelines for workload and evaluation.

This metric may be informative, but it is not the primary measure of blockchain performance. Systems will typically be deployed adjacent to the blockchain to facilitate significant reading and queries.

- Read latency (RL): Read latency is the time between when a read request is submitted (submit time, ST) and when the reply is received (response time, RT).

$$RL = RT - ST$$

- Read throughput (RT): Read throughput is a measure of how many read operations (total read operations, TRO) are completed in a defined time (whole time in seconds, TTS), expressed as reads per second (RPS).

$$RT = TRO / TTS$$

E. ANALYSIS OF COMPARING OF ZERO-KNOWLEDGE PROOF

The following shows the generation time of ZK-SNARKs and CL-Signature when the number of users increases and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

TABLE 5. Implementation system environment configuration

Spec	Root CA	I-CA	Verifier	DApp	User
CPU	Cascade Lake vCPU x 2	Cascade Lake vCPU x 2	Intel Core i7 3770K 3.5GHz	Cascade Lake vCPU x 2	Intel Core i5 8th Generation 1.4GHz
GPU	-	-	NVIDIA GeForce GTX960	-	Intel Iris Plus Graph-ics645
RAM	4GB	4GB	12GB	4GB	16GB
SSD	100GB	100GB	Samsung SSD 840 series 500GB	100GB	SSD 256GB
Python	Python 2.7.16 Python 3.7.4	Python 2.7.16 Python 3.7.4	Python 2.7.16 Python 3.7.4	Python 2.7.16 Python 3.7.4	Python 2.7.16 Python 3.7.4
Go	1.15.1	1.15.1	1.15.1	1.15.1	1.15.1
Docker	19.03.12	19.03.12	19.03.12	19.03.12	19.03.12
Node.js	12.18.4	12.18.4	12.18.4	12.18.4	12.18.4
HLF	HLF-samples 1.4.8	HLF-samples 1.4.8	HLF-samples 1.4.8	HLF-samples 1.4.8	HLF-samples 1.4.8
Number	1	2	1	1	3

the verifier is fixed to 1. Also, the maximum delay time, minimum delay time, average delay time, and throughput results are shown.

1) ZK-SNARKs Generation and Verification Comparison

As a comparison table of ZK-SNARKs generation and verification, it can be seen that the average latency is reduced, and the throughput is also increased if the user increases, as shown in Table 6.

2) CL-Signature Generation and Verification Comparison

As a comparison table of CL-Signature generation, if the user increases, such as shown in Table 6, the average latency decreases, and the throughput also increases. However, it can be seen that the average latency is slightly higher than that of ZK-SNARKs.

The graph in Figure 8 compares ZK-SNARKs and CL-Signature. As the number of users increases, the average delay time is less in ZK-SNARKs than in CL-Signature.

F. NUMBER COMPARISON BY NODE PARTICIPANT AND VERIFIER

When the numbers of users and verifiers increase by 1, the generation times of ZK-SNARKs and CL-Signature are shown. In addition, the maximum delay time, minimum delay time, average delay time, and throughput results are also shown in Figure 9.

1) ZK-SNARKs Generation and Verification Comparison

As shown in Table 7, if the number of users increases, the average delay time decreases, and the throughput increases. However, it can be seen that average delay time and throughput decrease as the number of verifiers increases.

2) CL-Signature Generation and Verification Comparison

As shown in Table 7, if the number of users increases, the average delay time decreases, and the throughput increases. However, average delay time and throughput decrease as the number of verifiers increases.

When the numbers of users and verifiers increase by 1, the average delay time of the CL-Signature is short, and the throughput is high. However, it was confirmed that ZK-SNARKs are more efficient when the number of users increases.

VI. ANALYSIS

A. SECURITY

- Anonymity: Our framework provides anonymity for a user, from the service provider and registration server. It means that the user can prove that authorize to use the registration server's services without revealing her identity to the registration server. **Proof** RS is attempting to hide the values r, t, x, and A from the user. The equations are sent to the registration server.

$$a = Q.r \tag{1}$$
$$b = (P1)x.Q.r \tag{2}$$
$$d = r.A \tag{3}$$
$$c = e(d, RC.R.(P2)t) \tag{4}$$

  The values of a, b, d, e, Q, R, $P1$ and $P2$ are known to the registration server [36]. The hardness of the elliptic curve Diffie-Hellman problem would be violated if the value of r was calculated from the first equation, $(P1)x$ from the second equation, A from the third equation, or $(P2)t$ from the fourth equation. As a result, determining values of $(r, (P1)x, A, (P2)x)$, let alone (r, t, x, A), is computationally impossible for the registration server. Due to the complexity of the inverse bilinear pairing operation problem (IBPOP), it is computationally infeasible to compute the second argument provided the fourth equation and values of c and d.
- Completeness: Since the input ϕ of $SimProve$ in the $Prove$ function is generated through the hash function and always gets the same output for the same input, always get the same ϕ for the correct data. According to Properties 1, $SimProve$, which proves that the input is ϕ and the output is π for the function $Prove$, is generated from the ZK-SNARKs algorithm $Prove$ that satisfies the completeness to ensure completeness. Because the π above is used as the output of $Prove$, the $SimProve\pi$ algorithm can also satisfy completeness.
- Computational Soundness: The inability to manipulate the proof π is based on the integrity of ZK-SNARKs used in the proposed Algorithm. The failure to drive ϕ used as the input of π depends on the integrity of the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

TABLE 6. ZK-SNARKs and CL-Signature comparison table

Scheme	User	Verifier	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (tps)
ZK-SNARKs	1	1	0.434	0.434	0.434	2.304
	5	1	0.425	0.034	0.117	20.736
	10	1	0.45	0.33	0.0934	20.968
	20	1	0.454	0.026	0.0387	26.078
CL-Signature	1	1	0.459	0.459	0.459	2.178
	5	1	0.438	0.038	0.204	19.612
	10	1	0.441	0.032	0.084	21.553
	20	1	0.447	0.026	0.055	26.42

tps : transaction per second

TABLE 7. ZK-SNARKs and CL-Signature comparison table (If user increases and verifier increases by 1)

Scheme	User	Verifier	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (tps)
ZK-SNARKs	1	1	0.434	0.434	0.434	2.304
	5	2	0.481	0.039	0.146	15.689
	10	2	0.467	0.3	0.087	22.852
	5	3	0.476	0.034	0.141	9.882
	10	3	0.506	0.032	0.103	20.503
CL-Signature	1	1	0.459	0.459	0.459	2.178
	5	2	0.47	0.038	0.142	16.25
	10	2	0.465	0.32	0.089	21.803
	5	3	0.485	0.044	0.159	12.937
	10	3	0.505	0.031	0.093	16.698

tps : transaction per second

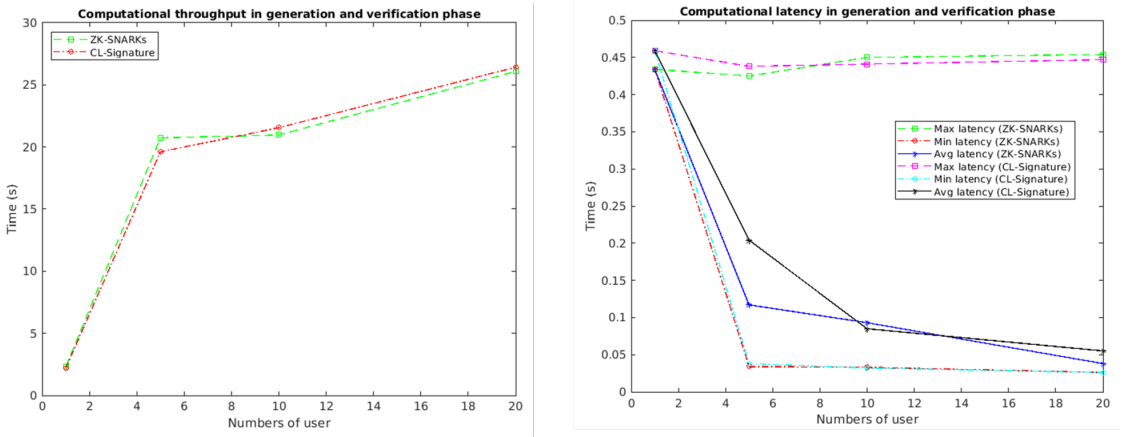

FIGURE 8. Comparison graph of ZK-SNARKs and CL-Signature

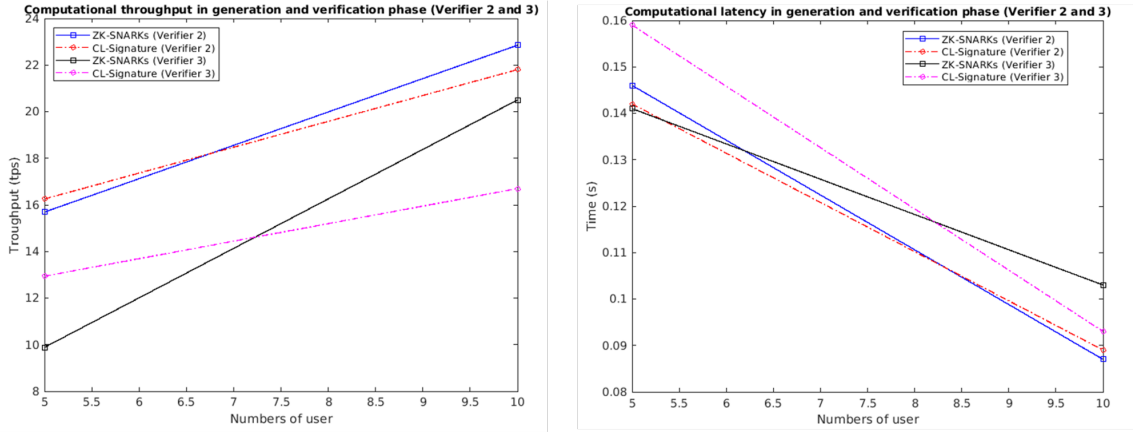

FIGURE 9. Comparison graph of ZK-SNARKs and CL-Signature (If user increases and verifier increases by 1)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3080329, IEEE Access

underlying blockchain. According to Properties 2, the input to the function $SimProve$ is ϕ, and π is generated from the ZK-SNARKs algorithm $Prove$ that satisfies the calculation proof to ensure computational integrity. Since pi is used as the output of $prove$, the $SimProve$ Algorithm also satisfies the computational soundness.

- Integrity: Imagine the case in which an attacker, α, attempts to distribute an updated upgrade to force the HIoT device to execute malicious code. It is impossible due to of several reasons. α should generate a ZK-SNARKs proof while posing as a trustworthy distributor. The proof must be generated by encrypting an upgrade file that matches the manufacturer's hash value released on the blockchain for it to be valid. A hub can consult the above at any time. As a result, whether the BC or the ZK-SNARKs system is not stable, and honest hub will never allow an unauthenticated file. Compromise of a hub gateway is the only other way to offer a malicious update. The protocol, on the other hand, is safe in this case. The HIoT system must check a signature on the hash value of the latest update made by the manufacturer before signing the identification challenge. The HIoT system will verify whether the update file matches the authenticated hash value in the protocol's final step after receiving it. As a result, such devices cannot accept malicious updates unless compromised.
- Privacy: Since the confidentiality of the transaction recorded in the blockchain is based on the selection of the encryption method to generate the cryptographic method, plaintext attack security (CPA-security), the probability of distinguishing between data1 and data2 containing other confidential information is meager (Besides, deriving personal information from the hash value recorded in the blockchain has a very low probability due to the oneway-ness of the hash used. Also, the probability of revealing the data value from the proof π generated by the user has a very low probability based on the zero-knowledge of Properties 3 of the proof algorithm used. Therefore, when considering the properties of ZK-SNARKs, there is a very low probability of revealing additional personal information values from cryptograms, identity attributes, and proofs. Therefore, the attack probability of data privacy of the protocol proposed in this paper has a negligible probability according to Properties 2.

B. EFFICIENCY

- When compared to existing schemes: When comparing the proposed system with the existing Shin et al. and Paul et al., VAIM presented the result shown in the following figure 10. Shin et al. use CL signature, indicating that the verification time is fastest when five users. However, we can see that VAIM is faster as the number of users increases. Also, Paul et al. can see that the verification time is similar to VAIM with five users, however slower than VAIM with more users. In other words, we can see that the more users there are, the more efficient the verification time of VAIM using ZK-SNARK.
- When the verifier is fixed to 1: As shown in Figure 8, if the number of users increases, the average latency of ZK-SNARKs and CL-Signature decreases, and the throughput increases. Users access the blockchain network using the application. As a result, the first user is connected to the blockchain network through the application, resulting in relatively high latency. However, since the second user does not proceed with accessing the blockchain network, it can be seen that latency is reduced and throughput is fast. The CL-Signature has a short average delay time and a high throughput.
- When the verifier increases by 1: As shown in Figure 9, if the number of users increases, the average latency of ZK-SNARKs and CL-Signature decreases, and the throughput increases. Similarly, latency and throughput are the same as ZK-SNARKs, with the first user accessing the blockchain network through the application, resulting in relatively high latency and low throughput. However, it can be seen that the number of verifiers increases, average delay time and throughput decrease. Also, when the number of users is small, the average delay time of CL-Signature is short, and the throughput is high, but when the number of users increases, it can be seen that ZK-SNARKs are more efficient.
  When there are many users, CL-Signature is slightly faster than ZK-SNARKs. However, as the number of verifiers and users increases, it can be seen that ZK-SNARKs are faster than CL-Signature.

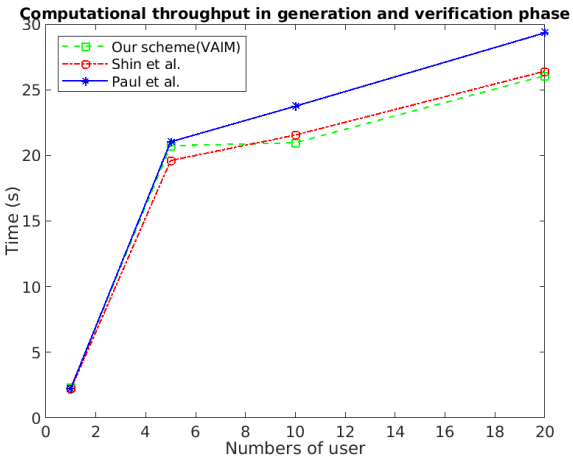

FIGURE 10. Comparison graph of existing schemes and VAIM

VII. CONCLUSIONS

Through HIoT, users are provided with beneficial and straightforward services through smart devices and remote servers. HIoT is equipped with practical and specific services

from a human-centric viewpoint but contains sensitive information. Therefore, network security is essential. This paper proposed an anonymous access control system for privacy preservation in the HIoT environment. ZK-SNARKs, BOMS, and blockchain-based protocols were built and developed. A blockchain network was built using HLF, and DApps were developed and provided to use them efficiently. We derived research results for various scenarios by researching and analyzing the throughput and delay time of NIZKs (ZK-SNARKs, CL-Signature). Comparison and analyses of ZK-SNARKs and CL-Signature confirmed that the average delay time decreased and throughput increased as the number of users increased. Also, it was confirmed that average delay time increased and throughput decreased as the number of verifiers increased. By contrast, according to differential research analyses of ZK-SNARKs and CL-Signature, when the number of users increases and the verifier is fixed to 1, the average delay time of CL-Signature is short, and throughput is high. Also, when the number of users increases by one and the number of users increases by one, the average delay time of CL-Signature is short, and throughput is high. Still, ZK-SNARKs are more efficient when the number of users increases. When there are many users, CL-Signature is generally slightly faster than ZK-SNARKs. However, as the numbers of verifiers and users increase, ZK-SNARKs are faster than CL-Signature.

If more than four verifiers exist, some peers (verifiers) will be down. In other words, if more than four peers are operating the HLF network, it cannot be processed and will be down. It indicates that building an I-CA to manage participants was more effective than the existing root CA managing all the participant entities on its own.

In the future, a protocol with enhanced specific security will be developed by applying it to a concrete environment. Continuous research to improve the speed of ZKP and blockchain performance is required. The code for implementing smart contracts has been made publicly available on GitHub.

REFERENCES

[1] U. Congress, "Andean strategy–hearing before the house select committee on narcotics abuse and control, 102nd congress, 1st session, june 11, 1991," 1991.

[2] A. Rghioui and A. Oumnad, "Internet of things: Surveys for measuring human activities from everywhere," International Journal of Electrical and Computer Engineering, vol. 7, no. 5, p. 2474, 2017.

[3] G.-A. B. M. C. Miranda, Mäkitalo and Murillo, "From the internet of things to the internet of people," IEEE Internet Computing, vol. 19, no. 2, pp. 40–47, 2015.

[4] P.-B. H. Silva, Zhang and Liebau, "People-centric internet of things," IEEE Communications Magazine, vol. 55, no. 2, pp. 18–19, 2017.

[5] G. Nitti and Atzori, "Trustworthiness management in the social internet of things," vol. 25, no. 6, pp. 1253–1266, 2013.

[6] A. K. Tyagi, G. Rekha, and N. Sreenath, "Beyond the hype: Internet of things concepts, security and privacy concerns," pp. 393–407, 2019.

[7] D. Todorov, Mechanics of user identification and authentication: Fundamentals of identity management. CRC Press, 2007.

[8] G.-B. Qin, Denker and Venkatasubramanian, "A software defined networking architecture for the internet-of-things," in In 2014 IEEE network operations and management symposium (NOMS). IEEE, 2014, pp. 1–9.

[9] P. Mahalle and Prasad, "Identity management framework towards internet of things," Ph.D. dissertation, 2013.

[10] Y. Yu, Y. Li, J. Tian, and J. Liu, "Blockchain-based solutions to security and privacy issues in the internet of things," IEEE Wireless Communications, vol. 25, no. 6, pp. 12–18, 2018.

[11] J. B. Bernabe, J. L. Canovas, J. L. Hernandez-Ramos, R. T. Moreno, and A. Skarmeta, "Privacy-preserving solutions for blockchain: Review and challenges," IEEE Access, vol. 7, pp. 164 908–164 940, 2019.

[12] D. Baars, "Towards self-sovereign identity using blockchain technology," Master's thesis, University of Twente, 2016.

[13] P. Dunphy and F. A. Petitcolas, "A first look at identity management schemes on the blockchain," IEEE Security & Privacy, vol. 16, no. 4, pp. 20–29, 2018.

[14] B. B. Patel, "Blockchain and digital signatures for digital self-sovereignty," Ph.D. dissertation, 2018.

[15] X. Zhu and Y. Badr, "Identity management systems for the internet of things: a survey towards blockchain solutions," Sensors, vol. 18, no. 12, p. 4215, 2018.

[16] A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum, "Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing," in Proceedings of the 14th ACM conference on Computer and communications security, 2007, pp. 276–285.

[17] Z. A. Lux, D. Thatmann, S. Zickau, and F. Beierle, "Distributed-ledger-based authentication with decentralized identifiers and verifiable credentials," in 2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS). IEEE, 2020, pp. 71–78.

[18] E. Androulaki, A. De Caro, M. Neugschwandtner, and A. Sorniotti, "Endorsement in hyperledger fabric," in 2019 IEEE International Conference on Blockchain (Blockchain). IEEE, 2019, pp. 510–519.

[19] S. Atapoor and K. Baghery, "Simulation extractability in groth's zk-snark," in Data Privacy Management, Cryptocurrencies and Blockchain Technology. Springer, 2019, pp. 336–354.

[20] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive zero knowledge for a von neumann architecture," in 23rd {USENIX} Security Symposium ({USENIX} Security 14), 2014, pp. 781–796.

[21] W. Jiang, H. Li, G. Xu, M. Wen, G. Dong, and X. Lin, "Ptas: Privacy-preserving thin-client authentication scheme in blockchain-based pki," Future Generation Computer Systems, vol. 96, pp. 185–195, 2019.

[22] Y. Li, G. Yang, W. Susilo, Y. Yu, M. H. Au, and D. Liu, "Traceable monero: Anonymous cryptocurrency with enhanced accountability," IEEE Transactions on Dependable and Secure Computing, 2019.

[23] A. Ishida, Y. Sakai, K. Emura, G. Hanaoka, and K. Tanaka, "Fully anonymous group signature with verifier-local revocation," in International Conference on Security and Cryptography for Networks. Springer, 2018, pp. 23–42.

[24] J. L. C. Sanchez, J. B. Bernabe, and A. F. Skarmeta, "Integration of anonymous credential systems in iot constrained environments," IEEE Access, vol. 6, pp. 4767–4778, 2018.

[25] S. M. Shin, K. Y. Lee, and K. J. Kim, "Modified anonymous attestation protocol of the tpm-installed device for the mobile environment," in CISC-W'09. KIISC, 2009.

[26] P. Mundhe, V. K. Yadav, A. Singh, S. Verma, and S. Venkatesan, "Ring signature-based conditional privacy-preserving authentication in vanets," Wireless Personal Communications, vol. 114, no. 1, pp. 853–881, 2020.

[27] D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox, "Zcash protocol specification," GitHub: San Francisco, CA, USA, 2016.

[28] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in 2016 IEEE symposium on security and privacy (SP). IEEE, 2016, pp. 839–858.

[29] A. Garoffolo, D. Kaidalov, and R. Oliynykov, "Zendoo: a zk-snark verifiable cross-chain transfer protocol enabling decoupled and decentralized sidechains," arXiv preprint arXiv:2002.01847, 2020.

[30] T. Koens, C. Ramaekers, and C. Van Wijk, "Efficient zero-knowledge range proofs in ethereum," ING, blockchain@ ing. com, 2018.

[31] Y. C. Tsai, R. Tso, Z.-Y. Liu, and K. Chen, "An improved non-interactive zero-knowledge range proof for decentralized applications," in 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON). IEEE, 2019, pp. 129–134.

[32] J. Groth, "On the size of pairing-based non-interactive arguments," in Annual international conference on the theory and applications of cryptographic techniques. Springer, 2016, pp. 305–326.

[33] J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," in International Conference on Security in Communication Networks. Springer, 2002, pp. 268–289.
[34] I. Ali, M. Gervais, E. Ahene, and F. Li, "A blockchain-based certificateless public key signature scheme for vehicle-to-infrastructure communication in vanets," Journal of Systems Architecture, vol. 99, p. 101636, 2019.
[35] J. K. Jiwon Lee, Jaekyoung Choi and H. Oh, "Saver: Snark-friendly, additively-homomorphic, and verifiable encryption and decryption with rerandomization," in IACR Cryptol. Cryptology ePrint Archive, 2019.
[36] X. P. Mehmood, Natgunanathan and Zhang, "Anonymous authentication scheme for smart cloud based healthcare applications," IEEE access, vol. 6, pp. 33 552–33 567, 2018.

GYEONGJIN RA is currently a Ph.D. student with the Department of Software Convergence at Soonchunhyang University (SCH), Asan, South Korea. Her research interests security and privacy protection, applied cryptography, blockchain. She received the M.S. degree in Computer Science and Engineering from Soonchunhyang University, Asan, South Korea, in Feb. 2018 and the B.S. degree in Computer Software and Engineering from the Soonchunhyang University, Asan, South Korea, in Feb. 2016. She has published papers on decentralized identity, blockchain, privacy at international conferences.

TAEHOON KIM was born in Seoul Republic of Korea, in 1995. He received the B.S. degree from the University of Soonchunhyang, Asan, in 2018. He is currently pursuing an M.S. degree with the Department of Computer Science and Engineering. His research interests information security, blockchain, key management, and privacy.

IMYEONG LEE is currently a Professor in the Department of Computer software engineering at Soonchunhyang University (SCH), Asan, South Korea. His research interests information security, cryptographic protocol, information theory, data communication. He received a Ph.D. degree in Information and Communication Engineering from Osaka University, Osaka, Japan, in 1989. He received an M.S. degree in Information and Communication Engineering from the Osaka University, Osaka, Japan, in 1986 and a B.S. degree in Electronic Engineering from Hongik University, Seoul, in 1981.