PAPER Special Section on Next-generation Security Applications and Practice

Identity Access Management via ECC Stateless Derived Key Based Hierarchical Blockchain for the Industrial Internet of Things

Gyeongjin RA[†a)], ***Student Member***, **Su-hyun KIM**[††b)], ***and*** **Imyeong LEE**[†c)], ***Nonmembers***

SUMMARY Recently, the adoption of the industrial Internet of things (IIoT) has optimized many industrial sectors and promoted industry "smartization." Smart factories and smart industries connect the real and virtual worlds through cyber-physical systems (CPS). However, these linkages will increase the cyber security danger surface to new levels, putting millions of dollars' worth of assets at risk if communications in big network systems like IIoT settings are left unsecured. To solve these problems, the fundamental method is security, such as authentication and confidentiality, and it should require the encryption key. However, it is challenging the security performance with the limited performance of the sensor. Blockchain-based identity management is emerging for lightweight, integrity and persistence. However, the key generation and management issues of blockchain face the same security performance issues. First, through blockchain smart contracts and hierarchical deterministic (HD) wallets, hierarchical key derivation efficiently distributes and manages keys by line and group in the IIoT environment. Second, the pairing verification value based on an elliptic curve single point called Root Signature performs efficient public key certificate registration and verification and improves the key storage space. Third, the identity log recorded through the blockchain is the global transparency of the key lifecycle, providing system reliability from various security attacks. Keyless Signature Infrastructure (KSI) is adopted to perform efficiently via hash-based scheme (hash calendar, hash tree etc.). We analyze our framework compared to hash-based state commitment methods. Accordingly, our method achieves a calculation efficiency of O(n log N) and a storage space saving of 60% compared to the existing schemes.

key words: *identity access management, IIoT, blockchain, keyless signature infrastructure*

1. Introduction

A CPS uses the Internet of Things and other networks to process tasks and information from the physical world in virtual space, adapting to changes without the need for human involvement [1]. In addition, it controls physical systems in real-time through a network based on high reliability and provides real-time and reliable data communication between systems [2]. Furthermore, it is noticed that many industries have started using this technology for enriching the industrial environments with advanced technologies, that lead to establishing what is known as Industrial Internet of Things (IIoT). IIoT devices, as any other IoT devices, consist of physical components, such as microcontrollers, transceivers, and memory. In addition to that, they are integrated with a set of simple protocols, which establish communication between IoT devices and their users, and written codes for managing and controlling the IoT devices [2]. However, IIoT devices differ in terms of their needs and requirements. In industrial fields, there are security requirements, hence there is a need of having security protocols for providing confidentiality, integrity, and authenticity for these devices and their data. Currently, the security level in IIoT devices is low. The lack of a robust key management systems, efficient identity authentication, low fault tolerance and many other issues lead IoT devices to being easily targeted by attackers [1], [3]–[5]. The most important thing to solve this problem is the encryption key for security, such as authentication and confidentiality. Simultaneously, lightweight mechanisms to efficiently manage numerous physical devices are essential. To this end, identity management is the first significant gateway for encrypted communication that performs device identification and key management, authentication, and access control through keys [4]–[7]. Figure 1 is a diagram of identity management for device and secure CPS data. Identity management performs device identification and authentication through public key certificates' validity and provisions them to various systems. Therefore, a public key generation and certificate issuance system is essential. Research has been conducted to solve the existing key distribution/escrow system and key computation and certificate issues [8]. Recently, a public key certificate system generated and manages lightweight keys using blockchain and

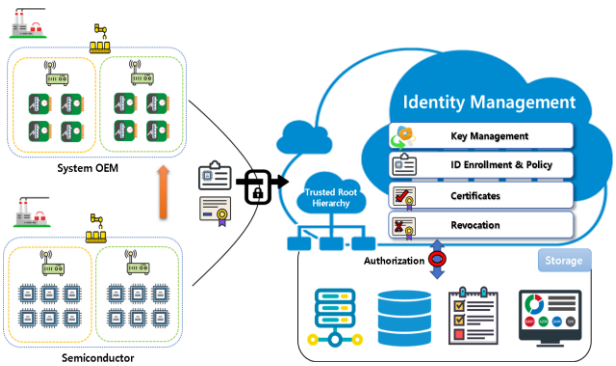

Fig. 1 Identity access management for IIoT

Manuscript received February 6, 2022.
Manuscript revised June 4, 2022.
Manuscript publicized July 28, 2022.

[†]The authors are with Dept. of Software Convergence, Soonchunhyang University, Asan KS002/31538, Republic of Korea.
[††]The author is with National IT Industry Promotion Agency, Jincheon 27872, Republic of Korea.
a) E-mail: rababi@sch.ac.kr
b) E-mail: ksh34@nipa.kr
c) E-mail: imylee@sch.ac.kr (Corresponding author)
DOI: 10.1587/transinf.2022NGP0003

elliptic curve encryption, and the hash algorithm has been researched [9], [10]. Blockchain key generation consists of non-deterministic and deterministic algorithms and hierarchical deterministic schemes. The generation of non-deterministic and deterministic keys based on the hash algorithm is efficient and improves privacy by deriving a single key through the hash's irreversibility [11]. However, it needs to remain a state, and there is much overhead for renewal and revocation. Since then, Bitcoin's BIP32 hierarchical key generation does not store the entire state through the elliptic curve's points, and since the upper layer derives the key to the lower layer, it efficiently manages each group of production lines such as IIoT [12]. However, a problem arises where privilege elevation attacks and privileges can be concentrated in one place. [13] binds and validates the public key's identifier and public key. It reduces communication overhead such as server reliability and synchronization through an immutable blockchain ledger.

Contribution: We propose a framework for new key generation and device identity management (IDM). Our research proposes an evolved method as a follow-up to the traditional research. First, hierarchical key derivations through blockchain smart contracts and hierarchical deterministic (HD) wallets efficiently distribute and manage keys by line and group in an IIoT environment. Second, we apply a pairing verification value based on a single point of an elliptic curve called Root Signature and a calculation through homomorphism. Unlike deterministic algorithms, this stateless commitment performs efficient public essential certificate registration and verification and improves the key storage space. It protects against privilege escalation and privilege concentration attacks by the upper layer performing only the key generation random seed of the lower layer. Third, it manages the binding of keys and identifiers through the blockchain. Stored records provide global transparency of the key life cycle, providing system stability against various security attacks. Our method presents a stateless vector commitment blockchain-based identity management framework using existing elliptic curves and cryptographic tools. Therefore, we analyze our framework by comparing it with the existing hash-based state method. As a result, our method achieves the computational efficiency of O(n log N) and a storage space saving of 60% compared to the conventional method.

Organization: The paper is organized as follows. Chapter 2 covers related research, which describes the key generation and certificate systems and elliptic curve cryptography. Chapter 3 introduces security requirements and mathematical knowledge. Chapter 4 details our scheme. Chapter 5 analyzes our scheme's security and efficiency, and Chapter 6 concludes.

2. Related Work

This chapter explains related work on identity management and key management wallets for the Industrial Internet of Things.

2.1 Identity Access Management

Identity access management is growing to include sensors, actuators, and smart devices, given the rapid expansion of the Internet of Things. It's more about giving access to IoT services and apps, as well as monitoring sensors, in this job. Many studies have been conducted in the field of identity management of things, and many models and frameworks have been proposed to support the rapidly expanding IoT network. A traditional X.509 public key infrastructure (PKI) approach is highly dependent on the vendor-provided list; thus, there is a single point of possible failure [6]. Accountable key infrastructure (AKI) can be used for public key validation via accountable internet protocol (AIP) self-authentication [7]. Using an integrity log server (ILS), the domain owner sends the certificate to a public log server, ensuring that the operation of a trust-based certificate authority (CA) is transparent and reliable. The distributed ledger features a global time stamp and a blockchain allowing real-time synchronization, thus reducing any need for a certificate revocation list (CRL) or an online certificate status protocol (OCSP), a CA-centric PKI featuring one-off recording.

Lo et al. [12], in contrast to the centralized method, evaluated a number of research on identity management models utilizing blockchain technology. Few publications proposed identification models to manage things, according to their review, whereas the majority of research employed the PKI technique for implementation. They also argued that the blockchain-based identification models they examined were not developed enough to support the IoT network. Dorri et al. [13] looked at an example of utilizing a blockchain in a smart house, where high-resource devices were deemed miners, in charge of managing all intelligent home communication and keeping the private blockchain. The framework's security, integrity, and availability were all tested as part of the project.

In contrast to the centralized method, [12], [13] evaluated a number of papers that used blockchain technology to examine identity management models. They offer a semi-decentralized Blockchain-based IoT identity management system with capabilities such as identity generation and ownership transfer, as well as identity portability across networks visited by devices. It also defines a collection of smart contracts that perform the registrar and management contract responsibilities. Existing methods, on the other hand, do not address the security and privacy issues that occur when devices from various administrative IoT domains (for example, factories) collaborate.

Meng et al. proposed an identity management system that ensures that authorized devices stay anonymous. Furthermore, session keys are negotiated between two parties, which can safeguard subsequent interactions. Ra et al. [15] proposed a KSI-based smart home secure communication scheme, a three-layer blockchain-based IoT smart home security architecture that includes blockchain, authentication and application layers with numerous characteristics

Fig. 2 Non-deterministic key generation

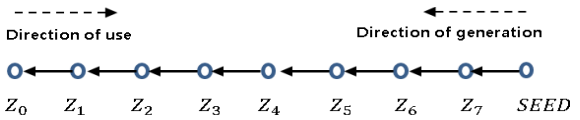

Fig. 3 Deterministic key generation

including identity authentication, privacy preserving and access control. KSI uses a single-use key for user authentication and generates a server-based global timestamp and blockchain for transaction records [16]. Keyless uses a key featuring an iterative and deterministic hash chain. However, its overhead storages commitment keys and the hash function do not provide encryption [17].

2.2 Key Management Wallet

A wallet is a place to store private and public keys to prove ownership (identity or data). The key generation method X categorizes non-deterministic, deterministic, and hierarchical deterministic key generation [19]. Non-deterministic key generation refers to the generation of a randomly selected public key pair by random seed (Fig. 2) [20]. If all keys are not backed up, recovery is not possible if the keys are lost. Therefore, the deterministic wallet continuously generates private keys through the one-way function from the randomly generated common random seed to compensate for the non-deterministic wallet's disadvantages (Fig. 3) [21]. The value before hashing is used to create the final irreversible value. The private/public key combination serves as the following private key's public key. The Hash function produces the same output value if the input values are the same, so get the common random seed to restore all consecutive private keys (Eqs. (1)-(2)). Every Z_i is a one-time-key. A one-way function renders it challenging to obtain a general solution.

$$Z_i = h(SEED) \text{ and use } Z_{(i-1)} = h(Z_i) \text{ as the public key} \quad (1)$$

$$Z_{i-1} = h(Z_i) \text{ (for all i = SEED, . . . , 1)} \quad (2)$$

In contrast, a trapdoor function is readily solved if secret information, such as an elliptic curve cryptography (ECC) or Rivest-Shamir-Adleman (RSA) algorithm. A one-way hash chain performs a hash operation over random seed to create public and private keys. The use of a one-time key (one-time authentication) is unlike the implementation of a hash. The value before hashing is a private key that only the user can have because of the irreversibility of the hash. The private/public key combination serves as the public key for the following private key [19].

The Lamport-Diffie signature key serves as the initial hash-based key and generates a key using two values from a single matrix and two columns [16]. After that, Winternitz, Winternitz+ scheme further reduced the key size using the XOR (exclusive OR A logical operator that is true when only one of the two inputs is true) operation of the bit string [21]. Generally, a one-way, function-based one-time key features a vector commitment data structure that efficiently verifies a public key composed of several private keys. The merkle signature scheme (MSS) uses a hash tree to construct a public key as a leaf node and employs the final root value as the verification value [22]. The extended MSS (XMSS) reduces the entire hash tree's overall storage overhead by halving the XOR operation of the bit string (Fig. 4) [23]. [23] apply the hash calendar and pebbling algorithm to improve the hash tree's verification and storage space efficiently. However, state vector commitment still incurs the overhead of changing the entire structure to store and update the state's value.

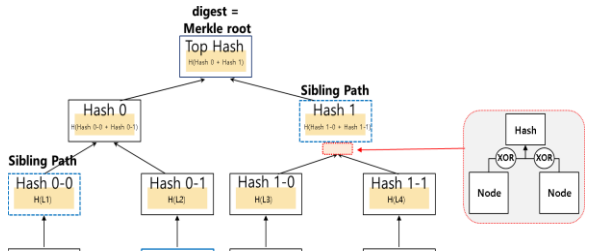

Fig. 4 Merkle hash tree & Extended Merkle hash tree

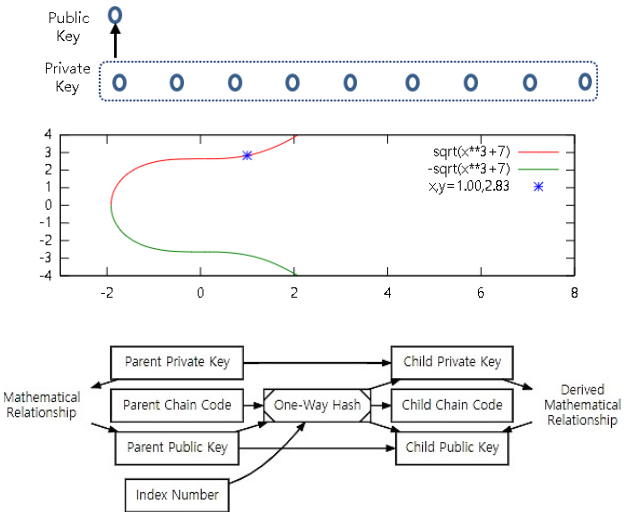

Fig. 5 Hierarchical deterministic key generation

The hierarchical deterministic wallet uses a point function, an elliptic curve digital signature algorithm (ECDSA) public key generation function that takes a large integer (private key) and converts it into a graph point (public key) (Fig. 5) [21]. It contains the keys generated in the tree structure, and the parent key can create a child key column, and each child key can create a grandchild key column. It means that two or more independent programs that agree on a

sequence of integers can no longer communicate and create a set of unique subkey pairs from a single parent key pair. It means stateless. HD wallet is made from root values of 128, 256, 512 size (bits) and generates a master private key and master chain code through HMAC-SHA512 algorithm [23]. However, in all existing deterministic wallets, including the Bitcoin Improvement Proposal 32 (BIP32) wallet, if the attacker has the upper public key and the lower private key, a privilege elevation attack that can quickly recover the upper private key occurs. Alkim. E. et al. [24] proposed dividing the public key into m to protect against attacks if maximum m private keys are not leaked with the master public key O(m) size. However, its nonefficiency is that it has to create a maximum of m private and public key pairs and is there are many key pairs to be stored. Fen. C.I. et al. [25] provided non-connectivity between parent and child public keys for anonymity of user identity and high scalability for key extraction. In addition, they proposed a new HD wallet that provides signatures with a trap door hash function instead of directly providing a private key for signature. However, the need to find the collision value using the trap door hash function requires much computation. Each time a transaction occurs, the collision value needs continuously computed to generate the required signature. Therefore currently, BIP32 allows the private key to only derive from the parent's private key to the public key from the public key.

3. Preliminaries

This chapter describes the security vulnerabilities of digital signature/public key systems and our scheme's security features.

3.1 Elliptic Curve Cryptography

A public-key cryptosystem is intractable; computational complexity theory indicates that computation can be concluded within a finite time, but this is too long in practice. ECC is a public key cryptosystem based on the elliptic curve theory. An elliptic curve is a curve defined by an equation of the form $y^2 = x^3 + ax + b$, yielding points on an ellipse, which contains a particular category of real numbers used to develop public keys. Early public-key cryptography was since it takes a very long time to divide a large integer into two or more prime numbers [24]. Elliptic curve cryptography exploits the fact that it also takes a very long time to determine the discrete logarithm of a random elliptic curve at any given point. Discrete logarithms require that x satisfies the equation $a^x = b$. If a and x are known, it is elementary to find b, but if only a and b are known, the difficulty of finding x is known as the discrete logarithm problem, exploited to render it very difficult to generate and use a public/private key [25]. In detail, the discrete logarithm problem can be stated as follows: it is difficult to find the logarithm of a surplus system that considers only the remainder of a modular operation. The advantage of ECC compared to conventional public key cryptographies such as RSA or Elgamal is that ECC affords a similar level of security but uses shorter keys.

3.2 Keyless Signature Infrastructure

KSI engages in repeated hashing to derive keys for random seed. Therefore, a key management system that efficiently validates multiple one-time keys ensures their authenticity is required. KSI uses binary hash trees to these ends. In other words, the public key is paired with the root value of the hash tree (using the leaf node approach) and with the last value of the hash chain [18]. It is thus possible to verify the final root value using only the nodes of the $\log n$ (n = leaf node number); the leaf node and the hash path are characteristic of the hash tree. Then, the chain is followed back stepwise to the initial creation time; the user employs the value before hashing as the private key, where only the user knows the hash's characteristics. At this time, the public value becomes the private key's public key. However, a hash chain is vulnerable to omnidirectional safety when the central part is exposed because the chain is linear; also, the use of a hash tree requires the entire node's storage to allow for root path calculations. KSI server creates a hash tree using information received from many users as leaf nodes. The KSI is hierarchical; the gateway collects lower-level client information and sends it to an aggregator and a link to global time. The publisher then adds the timestamp stored in the hash calendar structure to a blockchain. Initially, the KSI used linked time stamping to prevent forgery; the data were available in public media or newspapers [18]. KSI is a blockchain layer generating unique times for specific user's message interactions in the form of messages; the signature function is recorded in the distributed general ledger and thus cannot be forged [18].

3.3 Bilinear Pairings

Bilinear pairing implies that solving the discrete logarithm issue is difficult when the multiplicative groups $\mathbb{G}1$ and $\mathbb{G}2$ of a larger multiplicative group n have the same constant p.

$\mathbb{G}1$'s constructor is $g1$, while $\mathbb{G}2$'s constructor is $g2$. ψ is the result of a homogeneous computation using $\mathbb{G}1$, $\mathbb{G}2$, and $\psi(g2) = g1$. e: $\mathbb{G}1 \times \mathbb{G}2 \rightarrow \mathbb{G}\tau$ in in $\mathcal{L} = (\mathrm{n}, \mathbb{G}1, \mathbb{G}2, \mathbb{G}\tau, e, g1, g2)$ is called a bilinear pairing if it meets the following requirements [26].

- **Bilinear:** $\forall \mathrm{u} \in \mathbb{G}1$, $v \in \mathbb{G}2$ and a, b $\in \mathbb{Z}_{\mathrm{n}}$: $e(u^a, v^b) = e(u, v)^{ab}$.
- **Nondegenerate:** for u $\in \mathbb{G}1$, $v \in \mathbb{G}2$, e(u, v) $\neq \emptyset$, where $\emptyset$ are the zero factors of $\mathbb{G}\tau$.
- **Computability:** For $\forall \mathrm{u} \in \mathbb{G}1, v \in \mathbb{G}2$, there is an efficient technique that allows the computation e(u, v)

3.4 Root Signature

Root signature is a concept based on an aggregator signature proposed by Yuan, M et al. scheme for anonymity at cryptocurrency [26]. After that, we modified and applied the

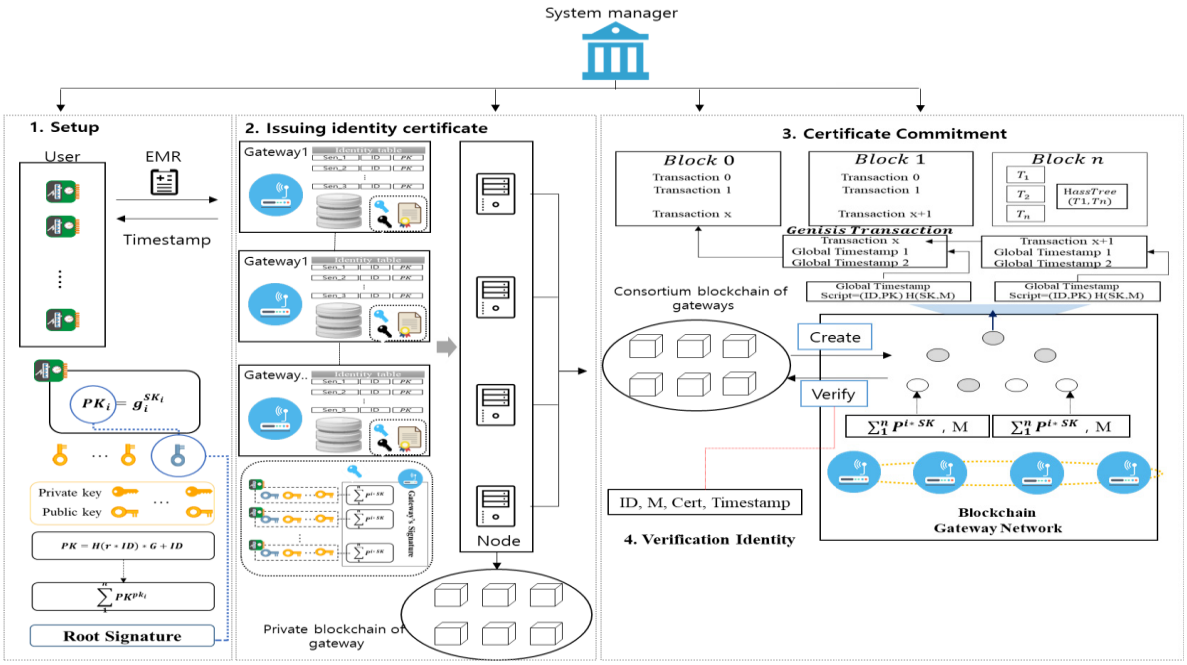

Fig. 6 The flow of our scheme via ECC stateless derived key sbased hierarchical blockchain

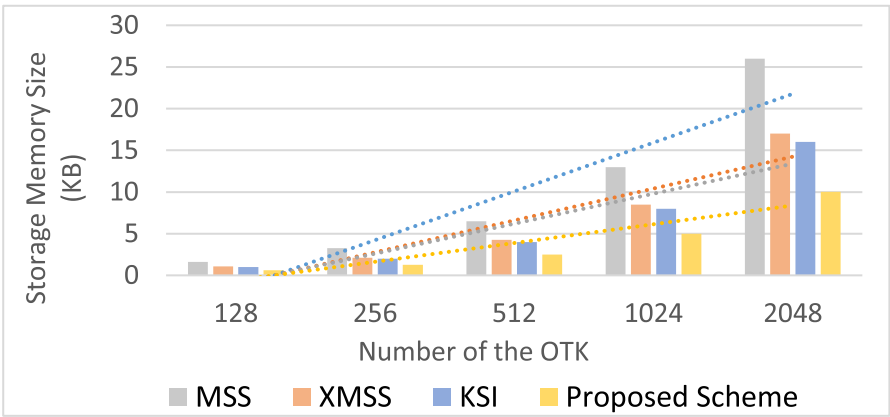

Fig. 7 Comparison of the graph memory size by number of OTK between existing scheme and proposed scheme

previously mentioned signature at permissioned blockchain membership management which was newly named "Root Signature" [27]. The four algorithms that make up a root signature are KeyGen, Sign, Combine, and Verify. The input to Algorithm Combine is a vector of n triples, each of which comprises a public key pk, message m, and signature σ. The method generates a single signature that may be used to sign all communications. An aggregate signature is what it's called, and it should have the same length as a signature on a single message. Finally, the Verify algorithm requires a vector of n pairings (pk, m) as well as a single aggregate signature as input. Only if were created as an aggregate of n valid signatures would it output "valid." We change the aggregate signature to generate a single validity value quickly. We modify the aggregate signature to efficiently create a single validity value of the one-time public key [28]. At that same time, the private one-time keys which yet secret need secret computation. We have been inspired by an aggregate signature's ECC compressibility and homogeneous bilinear pairings [26], [29]. A root signature is 3-tuple of algorithm 1 (KeyGen, Combine, and Verify) as follows:

Algorithm. 1 Root signature 3-tuple

(1) KeyGen (Public parameter) $\rightarrow$ $\{r_i \in Z_p^*, R_i = r_i{*}G, 1 <= i\}$: Using ECC, the user generates a one-time key pair (r_i, R_i). R_i refers to a one-time public key, whereas (r_i, R_i) is the public key set. Furthermore, each $R_i \subseteq R_i$ one-time public key is paired with a_i one-time private key.

(2) Combine $(R_i) \rightarrow (\sigma)$: User $u \in U$ computes $P = Hs(r_iR_i)G + R_i$ a public key verification value by computing a root signature $\sigma \leftarrow \sum_1^k A_i$, corresponds to the

one-time public key.

(3) Verify $(P, \sigma, a_i) \rightarrow \{0, 1\}$: $R = r_i{*}G\,? = 1$; otherwise $\perp$.

It can be called "stateless vector commitment". It saves storage one-time key and secrecy without the sensor's private key disclosure using ECC compressibility and homogeneous bilinear pairing. The user's one-time public key and root signature are accessible to others, but not the private key. Simultaneously, the root signature and the one-time public key have the same size (Fig. 7).

4. Proposed Scheme

In this chapter, we propose Identity management via ECC stateless derived key based KSI layer for IIOT environments. We consider system requirements, system design, and protocol.

4.1 System Requirements

The following security elements and efficiency need the ECC-based identity management framework system for the IIOT environment.

- **Confidentiality:** A message's confidentiality refers to the fact that it is not visible to other users. Encryption can be used to protect sensitive information that has to be kept private, as shown in [17]–[19]. The overhead of encryption and decryption is significant.
- **Digital Signature:** To join the permissioned blockchain network, users must first authenticate their identities. Access to the services is restricted to only authenticated users. Signatures may only be generated by users who have been verified by the permissioned blockchain. Only group members can verify signatures. Signatures that do not pass verification are considered invalid. At the same time, no communication should be vulnerable to forging or modification by a hostile attacker, and the signer can subsequently retract his or her signature.
- **Availability:** Even if an attack on a potential single-point-of-failure is detected, all users need to use legitimate services correctly and continuously through a distributed server.
- **Forward security:** Exposure of some or all keys should not affect subsequent authentication or signing.
- **Cross-certification:** When using server-based blockchain authorization, a legitimate server provides proxy signatures; these must be reliable, and mutual user/server authentication is essential.

Security attacks on identity management systems may be of the following types.

- **Man-in-the-middle (MITM) Attack:** An unsecured identity provider environment may intercept a message by a malicious attacker, change the public key, and authenticate and communicate using that public key; authentic participants remain ignorant of such behavior.
- **Replay Attack:** An unsecured identity provider environment may intercept a message by a malicious attacker and inappropriately retransmit the same message after a certain period to perform undesirable authentication and communication.
- **Single-point-of-failure:** Any collection of single servers is vulnerable to an attack (internal or external) on any server.
- **Key Search Attack:** Common signatures are signed using private keys and repeatedly employed, rendering them vulnerable to key analogy attacks.

4.2 System Design

Our scheme's entities consist of four components. (1) System manager is who issues security and computation parameters, (2) The sensor is who requests a certificate to the gateway. (3) The gateway generates and issues a certificate to the sensor. (4) The gateway network based on KSI consists of each gateway, the host server, and sensor as the sender & signer, and another sensor as receiver & verifier who verify the sender's certificate. As shown in Table 1, the blockchain system features a full node (thus with both a block header and a body) and a lightweight node (block header only). Therefore, the server and network verify a full-node transaction, but the sensor uses simplified payment verification (a lightweight node) to calculate a hash path validating the transaction to the full node. The proposed scheme system parameters are in Table 2. Our scheme is composed of 4 phases (Setup, Issuing identity certificate, Certificate commitment, Verification identity) (Fig. 6). Each phase includes a 4-tuple of algorithm 2 (GenKey, Issue_Cert, Cert_Com, Query_Cert) as follows:

Algorithm. 2 Our system 4-tuple

(1) KeyGen $(pp, SK_i) \rightarrow (PK_i, \sigma, R)$: All entities (Sensors, Gateway) generate their one-time public/private key and compute the root signature for a single validation value of one-time keys.

- All entities (Sensors, Gateway) are received by the public parameters PP.
- All entities generate the one-time public/private key.
- All entities compute root signature via combine **Algoritm 1.**

(2) IssueCert $(P_i, R) \rightarrow$ Cert (ID, R, σ_G): The sensor requests to the gateway register its identity and public key. Then, the gateway stores the sensor's identity information to manage their group. Gateway computes whether the validity of the sensor's one-time key and root signature. If the sensor is verified, the gateway signs using its private key. The gateway network obtains a signature after verification and handles the certificate as a hash tree leaf node. The gateway returns certificates containing ID and verification σ_G certificate to the sensor. The certificate is verified by sensors and the gateway network, including the ID and verification σ_B through R.

Table 1 The entities of proposed scheme

Character	Gateway & Gateway Network	Sensor
Type	Full Node	Lightweight Node
Role	Public Key, Authentication, Timestamp verification	Public Key, Authentication, Request of timestamp verification
Block Generation	O	X
Block Verification	O	O
Transaction Verification	O	X

Table 2 System parameters of proposed scheme

Parameter	Description
s_i	Sensor, $s_i \in$ Set of sensor S
g_i	Gateway as an identity provider, $g_i \in$ Set of gateway network G
H	Hash function for cryptography $\left(\{0,1\}^* \times G \times G \to Z_q^*\right)$
H_1	Hash function for cryptography $\left(\{0,1\}^* \times \{0,1\}^* \times G \times G \to Z_q^*\right)$
E	Group of points on an elliptic curve
r	Random value $r \in [n-1]$
R	Random value for the public, $R = r * G$
ID_*	Identifier of *, $ID_{S*} \in E$
PK_{*i}	Public key of * $(1 \le i \le n)$, $PK_{*i}=g_2^{SK_{*i}}$
SK_{*i}	Private key of * $(1 \le i \le n)$, $SK_{*i} \leftarrow Z_p$
σ_s	Sensor's root signature through one-time public key
σ_g	Gateway's root signature trough cetificate of sensors
S_t	The hash tree and blockchain of the gateway network create a global timestamp token
t_0	Root signature creation time for one-time public key, $t_n = t_0 + i, (1 \le i \le n)$

Algorithm 1. Setup (KeyGen)

Input: $pp = G, P, q, p, PK_{Si}, H_1$

Output: $SK_{*i}, PK_{*i}, \sigma_*$

begin

compute $\xrightarrow{\$}$ $PP = (G, P, q, p)$

select $\xrightarrow{\$}$ $h_1 : \{0,1\}^* \to G_1, h_2 : \{0,1\}^* \to Z_p^*$

compute $\xrightarrow{\$}$ $PP = \{P, q, G, p, H_1, H_2\}$

$SK_{*i} \leftarrow Key\_Gen(pp)$, $c \in Z_q^*$

$PK_{*i} \leftarrow Key\_Gen(pp)$, $PK_{*i} = c \cdot P$

return PK_{*i}, SK_{*i}

Public parameters PP, PK_{*i}, kept hidden SK_{*i}

for $i = 1, \ldots, n$ **do**

$\sigma_* \leftarrow \sum_1^n P_i^{PK_{*i}}$

$P_i = H(r * ID_*) * G + ID_*$

end for

return σ_*

end

(3) CertCom (R, ID_c, Sig σ) → Block (Tx num): The gateway creates a blockchain with the signed certificate through the gateway network. Since the certificate cannot be modified after the block is agreed upon, its validity is guaranteed. The gateway sends the global timestamp and signature generated by the hash tree to the gateway network. The gateway network forms a hash tree through transactions and adds blocks through consensus.

(4) QueryCert (ID, i, sk_i, c_i, S_t) → True/False, Hash Path: The gateway verifies the information in the certificate through the gateway network's blockchain. The gateway verifies the certificate sent by the sensor, authenticates the sensor, and then receives the message. Verifies the certificate's validity via the gateway network's blockchain and hash path computed hash root value.

4.3 Protocol

4.3.1 Setup Phase (KeyGen)

This phase is that the client creates an ECC-based one-time public/private key pair. The root signature of the full one-time public key is then produced to secure the one-time public key's later integrity. The client then requests the certificate and an ID, a one-time public key, the root signature, and the timestamp created by the blockchain server. Assume security parameter (G, P, q, p) is used to process all systems.

Step 1. As indicated in Eq. (3), the sensor chooses a random number for $c \in Z_q^*$ computes the public parameter PP and generates the public key PK_{Si} and private key

$$\text{received } PP = \{P, q, G, p, H_1, H_2\} \to$$
$$SK_{si} = c \in Z_q^* \to PK_{Si} = c \cdot P \quad (3)$$

SK_{si}. At that same time, the private key is kept hidden when PK_{Si} and PP are made public.

Algorithm 2. Issuing identity certificate (IssueCert)

Input: $ID_*, \sigma_*, t_0, PK_{*i}$
Output: $True/False$
begin
compute$\xrightarrow{\$}$
$P_i' = H(R * ID_*) * G + ID_*, (\sigma_*, g_2) =? \prod_{i=i}^{K} e(P_I, PK_{*i})$
$ID_S * R = ID_S * r * G = ID_S * r$
$E(\sigma, g_2) = e\left(P^i, g_2^{SK_{*i}}\right) = (P_i', PK_{*i})$
if $Verif(R)$ **then return** $\perp$
end for
return $True/False$
else $break$
end if
end

$$P_i = H(r * ID_s) * G + ID_s \rightarrow \tag{4}$$

$$\sigma_s = \sum_{1}^{n} P_i^{PK_{si}} \tag{5}$$

Step 2. Sensor computes a root signature σ_S value allowing for the one-time public key.

Step 3. As indicated in Eq. (6), the gateway chooses a random number for $c \in Z_q^*$ computes the public parameter PP and generates the public key PK_{gi} and private key SK_{gi}. At that same time, the private key is kept hidden when PK_{gi} and PP are made public.

$$\text{received PP} = \{P, q, G, p, PK_{gi}, H_1\} \rightarrow$$
$$SK_{gi} = c \in Z_q^* \rightarrow PK_{gi} = c \cdot P \tag{6}$$

Step 4. The gateway also computes a root signature σ_G value allowing for the one-time public key as follows the equation

$$P_i = H(r * ID_G) * G + ID_G \rightarrow \tag{7}$$

$$\sigma_g = \sum_{1}^{n} P_i^{PK_{gi}} \tag{8}$$

4.3.2 Issuing Identity Certificate Phase (IssueCert)

In this phase, that certificate of identification is issuing; the blockchain server creates a certificate using the network. A private key signature is created as a single root signature and added to the client's certificate.

Step 1. The sensor sends information about certificates $(D_s, \sigma_s, t_0, PK_{si})$ and requests for certificates to the gateway.

Step 2. The one-time public key PK_{Si} of the certificate information given by the sensor is used by the gateway to verify R. If it follows Eq. (9), return 1, else 0.

$$\text{Compute}[P_i' = H(R * ID_S) * G + ID_S, (\sigma_s, g_2)]$$
$$=? \prod_{i=i}^{K} e(P_i, PK_{si}) \rightarrow \tag{9}$$

$$ID_S * R = ID_S * r * G = ID_S * r \rightarrow \tag{10}$$

$$E(\sigma, g_2) = e\left(P^i, g_2^{SK_{si}}\right) = (P_i', PK_{si}) = OK \tag{11}$$

Step 3. If the sensor is verified, the gateway signs using its own SK_{Gi}.

$$H(r * ID_g) * G + ID_g, \tag{12}$$

Step 4. The gateway generates certificate signature σ_B which information on multiple sensors.

$$P_i = H(r * ID_g) * G + ID_g \rightarrow \tag{13}$$

$$\sigma_g = \sum_{1}^{n} P^{i\,SK_{gi}} \tag{14}$$

Step 5. The server's signature is verified by the gateway network and the server. If it follows Eq. (15), return 1; otherwise, return 0.

$$\text{Compute}\,[P_i' = H(R * ID_g) * G + ID_g, (\sigma, g_2)] =?$$
$$\prod_{i=i}^{K} e(P_i, PK_{gi}), \tag{15}$$
$$ID_g * R = ID_g * r * G = ID_g * r, \tag{16}$$
$$E(\sigma, g_2) = e\left(P^i, g_2^{SK_{gi}}\right) = (P_i', PK_{gi}) = OK. \tag{17}$$

Step 6. The gateway network gathers a signature after verification and stores it as a hash tree leaf node.

Step 7. The gateway returns certificates $(ID_S, \sigma_s, t_0, PK_{si}, ID_g, \sigma_g)$ containing ID and verification certificate to the sensor.

Step 8. The certificate's ID_g and verification σ_g are checked by sensors. If it follows Eq. (18), return 1; otherwise, return 0.

$$\text{Compute}\,[P_i' = H(R * ID_s) * G + ID_g, (\sigma_s, g_2)]$$
$$=? \prod_{i=i}^{K} e(P_i, PK_{gi}) \tag{18}$$
$$ID_g * R = ID_g * r * G = ID_g * r, \tag{19}$$
$$E(\sigma, g_2) = e\left(P^i, g_2^{SK_{gi}}\right) = (P_i', PK_{gi}) = OK, \tag{20}$$

4.3.3 Certificate Commitment Phase (CertCom)

This phase is that certificate commitment; the sensor requests that the gateway network sign by sending the paired private key and the message value, and the public key certificate. The gateway generates timestamps within the gateway network. The gateway's timestamp signature is routed to the root signature included in the blockchain and returned to the sensor.

Step 1. To a signature, the gateway constructs the pair $x = H(\text{Certificate}, SK_{Gi})$. Along with certificate information, it contains $(ID_s, \sigma_s, t_0, PK_{si}, ID_g, \sigma_g)$, SK_{gi}.

Step 2. After establishing the authenticity of the sensor with a private key SK_{Gi} and a public key PK_{gi}, each gateway checks the validity of the public key certificate using the gateway network's hash tree. If it follows Eq. (21), successfully return 1, else 0.

Algorithm 3. Certificate commitment (CertCom)

Input: $x = H(\text{Certificate}, SK_{Gi})$
Output: S_t
begin
if $Verifyid(PK_{*i} =? g_2^{SK_{*i}}) = \text{True}$ **then**
$H(ID_A, PK_A, SK_{Gi}) \rightarrow x$
$\text{Hash Tree}[SK_{gi}, ID_s, PK_s] \leftarrow \text{Hash Root}(x)$
$\text{compute} \xrightarrow{\$} \text{Hash Root}(h^{w-1}(\sigma))$
for $i = 0, \ldots, w-1$ **do**
$\sigma[i+1, k] = H(\sigma[i, k] || \sigma[i, k+1])$
return S_t
else *break*
end if
end

Algorithm 4. Verification identity (QueryCert)

Input: $M, certificate, \text{token}$
Output: *Hash Path*
begin
$\text{compute} \xrightarrow{\$} (S_t, SK_{gi}) \rightarrow$
$\text{Hash Tree}[SK_{gi}, ID_s, PK_s] \leftarrow \text{Hash Root}(x)$
$\text{compute} \xrightarrow{\$} \text{Hash Root}(h^{w-1}(\sigma))$
for $i = 0, \ldots, w-1$ **do**
$\sigma[i+1, k] = H(\sigma[i, k] || \sigma[i, k+1])$
if $\text{Hash Root} \in ? \text{Block}_{num} = \text{True}$ **then**
return Hash Path= $\sigma[n+1, k] = H(\sigma[n, k] || \sigma[n, k+1]$
else *break*

$$PK_{Ai} =? g_2^{SK_{Gi}} \tag{21}$$

Step 3. The gateway network generates a global hash tree about the all sensor $s \in S$, using relationship $x = H(ID_A, PK_A, SK_{Gi})$ and then a global timestamp value S_t, by linking the global coordinated time to the Hash Root value. (w = number of Leaf node, n = number of left binary tree, k = number of right binary tree, i = 0, . . . , n)

$$S_t = \text{Hash Tree}[SK_{gi}, ID_s, PK_s] \rightarrow \text{Hash Root} \tag{22}$$

$$x = H(ID_s, PK_s, SK_{gi}) \rightarrow$$
$$\text{compute}\,[\text{Hash Root}(h^{w-1}(\sigma))] = \sigma[i+1, k] = \tag{23}$$
$$H(\sigma[i, k] \,\|\, \sigma[i, k+1])$$

Step 4. The gateway network broadcasts to the blockchain server and sensor after generating token(S_t, SK_{Gi}) via internal consensus.

4.3.4 Verification Identity Phase (QueryCert)

This phase is that when other sensors receive the sensor's unique timestamp, the public key certificate, and the sender's message, the other sensor verifier asks the blockchain network to validate the public key certificate and the timestamp. The server uses the distributed branching block and then passes the hash path to the verifier, reviewing and verifying validity.

Step 1. The sensor sends M = (mgs[i]), the public key certificate ($ID_s, \sigma_s, t_0, PK_s, ID_g, \sigma_g$), the private key SK_{si}, the token (S_t, SK_{gi}).

Step 2. The verifier requests the gateway network to verify the certificate and the token (S_t, SK_{gi}) of the sensor.

Step 3. The gateway network verifies the certificate's validity using a hash tree and returns a valid *Hash Path* to verify the verifier's token. The gateway network returns Hash Path = $\sigma[n+1, k] = H(\sigma[n, k] \,\|\, \sigma[n, k+1]$

$$(S_t, SK_{gi}) \rightarrow$$
$$\text{compute}\,[\text{Hash Root}(h^{w-1}(\sigma))] =? \sigma[n+1, k] \tag{24}$$
$$= H(\sigma[n, k] \,\|\, \sigma[n, k+1] \rightarrow$$
$$\text{Hash Root} \in ? \text{Block\_num} \tag{25}$$

5. Analysis of Our Scheme

In this section, we analyze the security and efficiency of our scheme.

5.1 Remarks and Related Work

Unlike the existing methods [1], [2], [4], our proposed method provides an IDM configured with a permissioned blockchain. Therefore, it is robust against many problems that arise from relying on CA. Man-in-the-middle attacks do not occur because multiple public and private keys are generated and registered. Also, because a unique timestamp value is attached to the certificate, replay attacks do not occur. Finally, it has immunity from single point of failure through distributed and hierarchical layer blockchain servers. In particular, the existing public key method [4], [13]–[15], [18] is dangerous for message stealing and manipulation after the private key is exposed. do. It is robust against speculative attacks of public and private keys through public key verification through bilinear pairings (Table 3 and Table 4).

5.2 Security

Our scheme considers various security threats and requirements. The environment is a KSI-based permission blockchain, which is more robust against MITM and replay attacks than identification providers, being more secure than a single PKI or AKI. Compared to other schemes, it offers the following security features (Table 3 and Table 4).

- **Confidentiality:** As encryption involves a transaction key generated via ECC, decryption is impossible even if a third party acquires a message; thus, the scheme is airtight. According to the Discrete Logarithm Problem (DLP), it is impossible to obtain a private key by setting

Table 3 The environment of other schemes and proposed scheme

	M.SC et al. [1]	BW. J et al. [2]	N. EM et al. [4]	Our scheme
Authentication/ Integrity	Offer	Offer	Offer	Offer
Identity management provider	CA RA	Integrity Log Server Validator	Server & Blockchain	
MITM attack	Weakness (CA High Dependency)	Weakness (CA Low Dependency)	Strength (Independency)	
Replay attack	Unsafe	Unsafe	Safe	
Single point of-failure	Unsafe	Unsafe	Safe	

Table 4 The security of other schemes and our scheme

	K. Gu [15]	D. Na et al. [13]	R. Lo et al. [14]	A. Bu et al. [18]	N. Em at al. [4]	Our scheme
Confidentiality	O		X		O	
Replay attack	△	◎	◎	◎	◎	◎
MITM-attack	△	◎	◎	◎	◎	◎
Cross-certification	O	O	O	X	X	O
Key search-attack	△	△	△	◎	◎	◎
Forward-security	△	△	△	△	△	◎

△: Weakness, ◎: Strength,
O: Offered, X: Not offered

the prime number p as a very large safe prime. Also, ECDLP is an elliptic curve version of DLP, and $PK_{Si} = c \cdot P$ is difficult to know, just as $SK_{si} = c \in Z_q^*$ is difficult to know.

- **Integrity:** A global timestamp is created by the gateway network when a certificate is typed and committed to the blockchain. The certificate is complicated to change, as all users would be required to synchronize their efforts to this end; thus, integrity is assured. For any coalition of fewer than n players, our scheme is player-private against an honest-but-curious adversary follow the DDH assumption. ((f, n)-DHE Problem): Let G be a group of prime order p, $h \in G$ and $a \in Z_p$. Given $h, h^a, \ldots, h^{a_n}$, output $(f(x), h^{f(a)})$, where $f(x) \in Z_p[x]$ is a polynomial function with $f(x) > n$. It is practically impossible to synchronize by manipulating [33].
- **Availability:** A distributed gateway network is not susceptible to attacks on a single point; constant availability is guaranteed.
- **Authentication:** Public-key certificates and key-pair $(PK_{Si}, SK_{Si}, \sigma)$ are used to identify and authenticate legitimate sensors; provided by Eq. (9)-(11): $ID_s * R = ID_S * r * G = ID_{sS} * r$, $E(\sigma, g_2) = e\left(P^i, g_2^{SK_{si}}\right) = (P_i', PK_{si}) = OK$
- **Non-repudiation:** Sensor denial is impossible; the gateway network creates a unique global timestamp for any certificate containing the sensor's one-time private key.
- **MITM attack:** A gateway network spreads an attack to a single gateway point across all points; a MITM attack is impossible. Provides by Eq. (15)-(17): $ID_g * R = ID_g * r * G = ID_g * r$, $E(\sigma, g_2) = e\left(P^i, g_2^{SK_{si}}\right) = (P_i', PK_{si}) = OK$,
- **Replay attack:** A gateway network distributes attacks to a single gateway point; replay attacks are impossible. Provides by Eq. (13)-(14), (22)-(23): ID_s, σ, t_0, PK_{si}
- **Single-point-of-failure:** A gateway network distributes attacks to a single point; It blocks

Table 5 BAN-logic notations

Notation	Description
$P\mid\equiv Y$	The formula Y can be believed by the entity P.
$P \lhd Y$	P sees Y.
$P\mid\Rightarrow Y$	P has achieved jurisdiction over Y.
$P\mid\sim Y$	P has once said Y.
$\#(Y)$	Y is fresh.
$\{Y\}_K$	Y is hidden under the secret K.
$P \leftrightarrow^K \theta$	P and θ establish a secret key K.

potential insider attackers. Provides by 4.4.3 Certificate commitment phase (Step 3-4): Using the relationship $x = H(Certificate, SK_{Gi})$, the gateway network creates a global hash tree, and then a global timestamp value S t by binding the global coordinated time to the root value. Next, the gateway network broadcasts to the blockchain server and sensor after aggregating $token(S_t, SK_{gi})$ via internal consensus.

- **Reliability:** A gateway network blocks potential internal attackers by insisting on internal consensus; arbitrary changes and deletions are rendered impossible by ensuring that all nodes follow the policy of using only proven transactions as blocks.
- **Key search attack:** As ECC-based keys are required for authentication and are not reused, the system is safe from attacks because the keys are not repeatedly exposed. Provides by Eq. (3)-(5): $SK_{si} = c \in Z_q^*$, $PK_{si} = c{\cdot}P$, $PP = \{P, q, G, p, PK_{si}, H_1\}$. $P_i = H(r{*}ID_s){*}G{+}ID_s$, $\sigma_A = \sum_1^n P^{i^{PK_{si}}}$ is then calculated as a value allowing for public key verification.
- **Forward security:** ECC-based, independent, one-time public/private key generation is employed. If a key hash value is disclosed, no future key based on that value can be computed to ensure omnidirectional security. Forgery of the sensor's key pair (PK_{si}, SK_{si}) is complicated based on the Elliptic Curve Discrete Logarithm Problem (ECDLP); Provided by Eqs. (4)-(5): $P_i = H(r * ID_g) * G + ID_g$, $\sigma_A = \sum_1^n P^{i^{PK_{gi}}}$
- **Cross-certification:** The gateway and all sensors engage in mutual authentication by handshaking a one-time secret value generated when creating a one-time public/private key and a root signature; Provided by Eq. (9)-(11), (15)-(17): $ID_g * R = ID_g * r * G = ID_g * r$, $E(\sigma, g_2) = e\left(P^i, g_2^{SK_{gi}}\right) = (P_i', PK_{gi}) = OK$

5.3 Logic Proof by Ban Logic

Burrows et al. [33] proposed a logic of authentication in 1989, which is popular in checking the correctness of authentication protocols. BAN logic is a belief-based model logic that can be used to prove whether the implementation of the protocol can achieve the expected goals and to discover shortages in the proposal design. The main notations are listed in Table 2. Based on the idea, our proposal considers the below logical rules in our proof.

- **A1. Message-Meaning Rule (MMR):** $\frac{P\mid\equiv P\leftrightarrow^K\theta, P\lhd\{Y\}_K}{P\mid\equiv\theta\mid\sim Y}$.
- **A2. Nonce Verification Rule (NVR):** $\frac{P\mid\equiv\#(Y), P\mid\equiv\theta\mid\sim Y}{P\mid\equiv\theta\mid\equiv Y}$.
- **A3. Freshness Propagation Rule (FPR):** $\frac{P\mid\equiv\#(Y)}{P\mid\equiv\#(Y,X)}$.
- **A4. Jurisdiction Rule (JR):** $\frac{P\mid\equiv(\theta\mid\Rightarrow Y), P\mid\equiv(\theta\mid\equiv Y)}{P\mid\equiv Y}$.

The method of using BAN logic for security proof is to infer from the security that the desired security target follows the four security assumptions given above.

- **Message 1**: $s_i \rightarrow g_i : (ID_*, \sigma_*, t_0, PK_{*i})$.
- **Message 2:** $g_i \rightarrow g_i : (H(ID_A, PK_A, SK_{Gi}) \rightarrow x)$.
- **Message 3:** $g_i \rightarrow s_i : S_t$.

Idealized form: The idealized form of our scheme is as below:

- **Message 1**: $s_i \rightarrow g_i : ID_s, \sigma_s, t_0, PK_{si}, ID_g, \sigma_g$.
- **Message 2:** $g_i \rightarrow g_i : PK_{Ai} =? g_2^{SK_{Gi}}$.
- **Message 3:** $g_i \rightarrow s_i$: M, *certificate*, token

In terms of our scheme description, we provide the below security hypothesises in our scheme.

- **H 1:** $s_i \mid\equiv \#(PK_{*i})$.
- **H 2:** $PK_{*i} \mid\equiv \#(PK_{si})$.
- **H 3:** $g_i \mid\equiv g_i \leftrightarrow^K \theta$.
- **H 4:** $PK_{Ai} \mid\equiv \#(g_2^{SK_{Gi}})$.
- **H 5:** $g_i \mid\equiv g_i \leftrightarrow^K S_t$.
- **H 6:** $S_t \mid\equiv \#(H(ID_A, PK_A, SK_{Gi})) \Rightarrow x \leftrightarrow^K S_t$.
- **H 7:** $S_t \mid\equiv \#(\text{HashRoot}) \Rightarrow h^{w-1}(\sigma) \leftrightarrow^K Block$.
- **H 8:** $Block \mid\equiv \#(S_t)$.

In addition, we provide the below security goals that aim to prove our scheme.

- **Goal 1.** $g_i \rightarrow g_i : PK_{Ai} =? g_2^{SK_{Gi}}$.
- **Goal 2.** $g_i \rightarrow s_i$: M, *certificate*.
- **Goal 3.** $S_t \mid\equiv \#(\text{HashRoot}) \Rightarrow h^{w-1}(\sigma) \leftrightarrow^K Block$.
- **Goal 4.** $Block \mid\equiv \#(S_t)$.

According to the Message 1 message, there is

$$SK_{*i} \lhd Key\_Gen(pp) \leftrightarrow^K c \in Z_q^*$$

We employ MMR and the assumption H1, this is:

$$PK_{*i} \mid\equiv Key\_Gen(pp) \leftrightarrow^K PK_{*i} = c \cdot P$$

We use the FPR, NVR and the assumption H7, this is

$$\sigma_* \mid\equiv \sum_1^n P_i^{PK_{*i}} \leftrightarrow^K P_i = H(r * ID_*) * G + ID_*$$

According to above equation and H4, we employ NVR, this is

$$[P_i'| = H(R{*}ID_S){*}G{+}ID_S, (\sigma_s, g_2)] \leftrightarrow^K \prod_{i=i}^{K} e(P_i, PK_{si})$$

In addition, according to above equation and the assumption H7, we use JR, this is:

Table 6 The key memory sizes of other schemes and our scheme

	R. C. Me et al. [8]	Alkim. E et al. [7]	Fan. C. I et al. [25]	J. Bu et al. [9]	A. Hu et al. [10]	Our scheme
Key Memory Size	(256 bits *4) * n	256 bits * n/2	256 bits * n	256 bits * 2 *n	128 bits * 2 * n	160 bits * 2 * n
Calculation Operator	One-way function					Elliptic curve

n: Number of the key
unit: bit

Table 7 The key verification memory sizes of other schemes and our scheme

	R. C. Me et al. [11]	NY.J. Bu et al. [12]	A. Bu et al. [18]	Our scheme
Memory Size	$256 * (2^{d+1}\text{-}1) + (160 * 2) * 2^{d}$	$128 * (3*2^{d}\text{-}2) + 160 * 2^{d}$	$256 * (2^{d+1}\text{-}1)$	$320 * (1+2^{d})$
Calculation operator	Elliptic curve + Hash function			Elliptic curve + Root signature

d: Dept of the hash tree
unit: bit

Table 8 The total memory size by number of one-time keys

	R. C. Me et al. [11]	NY.J. Bu et al. [12]	A. Bu et al. [18]	Our scheme
Number of the OTK	Memory Size (KB)			
128	13	8.5	8	5
256	26	17	16	10
512	52	34	32	20
1024	104	68	64	40
2048	208	136	128	80

$$ID_S * R \mid = ID_S * r * G \mid \leftrightarrow^K ID_S * r$$

From the Message 2 message, there is

$$\sigma_g \leftrightarrow^K \sum_{1}^{n} P^{i^{SK_{gi}}}$$

We use MMR and the assumption H3, this is:

$$[P_i' \mid \equiv H(R * ID_s) * G + ID_g, (\sigma_s, g_2)] \leftrightarrow^K \prod_{i=i}^{K} e(P_i, PK_{gi})$$

According to H2 and above equation, we employ FPR, this is

$$ID_g * R \mid \equiv ID_g * r * G \leftrightarrow^K ID_g * r,$$

By above equation, we use NVR, this is

$$H(ID_A, PK_A, SK_{Gi}) \leftrightarrow^K x$$

According to the Message 3 message, there is'

$$\text{Hash Tree}[SK_{gi}, ID_s, PK_s] \mid \equiv \text{Hash Root}(x) \leftrightarrow^K \text{Hash Root}(h^{w-1}(\sigma))$$

and above equation via MMR, this is:

$$\sigma[i+1, k] \mid \equiv H(\sigma[i, k] \,||\, \sigma[i, k+1])$$

Then, we employ FPR, this is

$$S_t \mid \equiv \#(H(ID_A, PK_A, SK_{Gi})) \Rightarrow x \leftrightarrow^K S_t.$$

According to above equation and H8, we use JR, this is

$$s_i \mid \equiv \#(ID_*, \sigma_*, t_0, PK_{*i}) \leftrightarrow^K S_t$$

Thus, the proof of the goals are achieved according to H3, above equation.

5.4 Efficiency

The proposed scheme generates a root signature during ECC-based, one-time public/private key generation; this is used to efficiently validate a single using the public key (Table 6). Our scheme's key memory size is the same as that of ECC (security ratio, 160 bits), more extensive than that of the Winternitz+ scheme but smaller than that of the hash

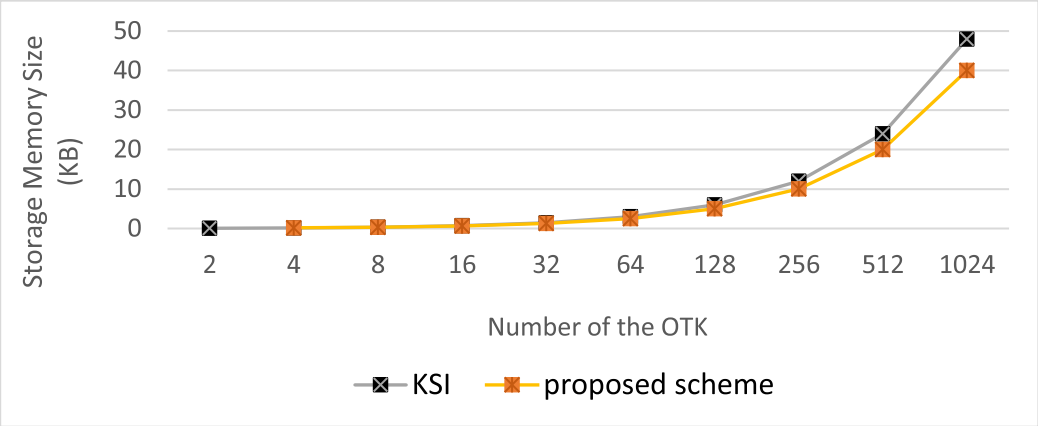

Fig. 8 Comparison of the graph memory size by number of OTK between KSI framework and proposed scheme

chain of KSI. However, the memory required to change and verify all one-time public keys is the lowest of all systems. As the MSS uses an existing Merkle tree, key sizes can be calculated. XMSS performs an XOR operation when concatenating a hash using an extended Merkle tree, doubling the overall safety in collision resistance and the birthday paradox; the memory size is halved. KSI computation is rendered efficient (thus requiring less memory) by employing the "padding traveling" algorithm. Our scheme generates a single compact root signature for all one-time public keys. The memory required is less than the hash function based on the sibling key generation scheme (Table 7). Especially, our scheme is an overhead to save close to 60% compared to other single using & authentication schemes (Table 8 and Figs. 7, 8).

6. Conclusion

Recently, the adoption of the industrial internet of things (IIoT) has optimized many industrial sectors and promoted industry "smartization." Smart factories and smart industries connect the real and virtual worlds through cyber-physical systems (CPS) [30]. However, these linkages will increase the cyber attack surface to new levels, putting millions of dollars in assets at risk if communications in massive network systems like IIoT settings are left unguarded [31], [32]. We developed an electronic signature and public key system for a blockchain using an elliptic curve cryptography-based key wallet, root signature, and digital technology. Furthermore, we compared the safety and efficiency of our scheme with those of existing systems. Our proposed scheme has effectively reduced the stored key overhead and is secure with mutual reliability.

We propose a framework for new device identity management and key generation. First, through blockchain smart contracts and hierarchical deterministic (HD) wallets, hierarchical key derivation efficiently distributes and manages keys by line and group in the IIoT environment. However, it requires much computation in that it has to create a maximum of m private and public key pairs and is not efficient because there are many key pairs to be stored. Currently, BIP32 allows the private key to only derive from the parent's private key to derive the public key from the public key [33]. Other approaches provide no connectivity between parent and child public keys for user identity privacy and high key extraction scalability. In addition, we propose a new HD wallet that provides signatures with a trap door hash function instead of directly providing a private key for signature. However, the need to find the collision value using the trap door hash function requires much computation. Each time a transaction occurs, the collision value needs continuously computed to generate the required signature.

Second, the pairing verification value based on an elliptic curve single point called Root Signature performs efficient public key certificate registration and verification and improves the key storage space. Third, the identity log recorded through the blockchain is the global transparency of the key lifecycle, providing system reliability from various security attacks. In particular, KSI's hash tree is adopted to perform efficiently. We analyze our framework compared to hash-based state commitment methods. Accordingly, our method achieves a calculation efficiency of $O(n \log N)$ and a storage space saving of 60% compared to the existing schemes. Therefore, our proposed scheme is safer and more efficient than the conventional scheme.

It can be applied to a network environment that safely identifies and manages sensors in various IIoT environments to manage production lines or groups in a general CPS environment. In the future, we will explore how our scheme functions in various similar environments. Furthermore, it is expected that the security and efficiency of the entire CPS will be improved by performing secure data communication and authentication.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government

(MSIT) (No. 2022R1A2B5B01002490), and Soonchunhyang University Research Fund.

References

[1] X. Liu, Y. Zhu, Y. Ge, D. Wu, and B. Zou, "A secure medical information management system for wireless body area networks," KSII Transactions on Internet and Information Systems (TIIS), vol.10, no.1, pp.221–237, 2016.

[2] X. Li, J. Niu, J. Gao, and Y. Han, "Secure electronic ticketing system based on consortium Blockchain," KSII Transactions on Internet and Information Systems (TIIS), vol.13, no.10, pp.5219–5243, 2019.

[3] N. Emmadi and H. Narumanchi, "Reinforcing Immutability of Permissioned Blockchains with Keyless Signatures' Infrastructure," Proceedings of the 18th International Conference on Distributed Computing and Networking ACM, p.46, 2017.

[4] D. Hankerson, A.J. Menezes, and S. Vanstone, Guide to elliptic curve cryptography, Springer Science & Business Media, 2006.

[5] J.W. Jang, S. Kwon, S. Kim, J. Seo, J. Oh, and K.-H. Lee, "Cybersecurity framework for IIoT-based power system connected to microgrid," KSII Transactions on Internet and Information Systems (TIIS), vol.14, no.5, pp.2221–2235, 2020.

[6] D. Chen, G. Chang, and J. Jia, "AC4E: An access control model for emergencies of mission-critical cyber-physical systems," KSII Transactions on Internet and Information Systems (TIIS), vol.6, no.9, pp.2052–2072, 2012.

[7] BW. Jin, J.O. Park, and M.S. Jeon, "A study on authentication management and communication method using AKI based verification system in electronic helath records environment," Journal of IIBC, vol.16, no.6, pp.25–31, 2016.

[8] R.C. Merkle, "A digital signature based on a conventional encryption function," In Conference on the theory and application of cryptographic techniques, pp.369–378, Springer, 1987.

[9] J. Buchmann, E. Dahmen, S. Ereth, A. Hülsing, and M. Rückert, "On the security of the Winternitz one-time signature scheme," In International Conference on Cryptology in Africa, pp.363–378, Springer, 2011.

[10] A. Hülsing, "W-OTS+–shorter signatures for hash-based signature schemes," In International Conference on Cryptology in Africa, pp.173–188, Springer, 2013.

[11] R.C. Merkle, "A certified digital signature," In Conference on the Theory and Application of Cryptology, pp.218–238, Springer, New York, 1989.

[12] NY.J. Buchmann, E. Dahmen, and A. Hülsing, "XMSS-a practical forward secure signature scheme based on minimal security assumptions," In International Workshop on Post-Quantum Cryptography, pp.117–129, Springer, 2011.

[13] D. Basin, C. Cremers, T.H.-J. Kim, A. Perrig, R. Sasse, and P. Szalachowski, "ARPKI: Attack resilient public-key infrastructure," In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pp.382–393, ACM, 2014.

[14] R. Longo, F. Pintore, G. Rinaldo, and M. Sala, "On the security of the Blockchain Bix Protocol and Certificates," In 2017 9th International Conference on Cyber Conflict (CyCon), pp.1–16, IEEE, 2017.

[15] K. Gu, N. Wu, Y. Liu, F. Yu, and B. Yin, "WPKI Certificate Verification Scheme Based on Certificate Digest Signature-Online Certificate Status Protocol," Mathematical Problems in Engineering, 2018.

[16] G.-J. Ra and I.-Y. Lee, "A study on KSI-based authentication management and communication for secure smart home environments," KSII Transactions on Internet and Information Systems (TIIS), vol.12, no.2, pp.892–905, 2018.

[17] T. Chow, W. Wong, Y. Rouskov, K.W. Chan, W. Jiang, C. Chow, and A. Belur, "Short-lived certificate authority service," US Patent No.7,853,995, 2010.

[18] Y.-C. Hu, M. Jakobsson, and A. Perrig, "Efficient constructions for one-way hash chains," In International Conference on Applied Cryptography and Network Security, pp.423–441, Springer, 2005.

[19] A. Buldas, A. Kroonmaa, and R. Laanoja, "Keyless signatures' infrastructure: How to build global distributed hash-trees," In Nordic Conference on Secure IT Systems an Introduction to Signal Detection and Estimation, New York: Springer-Verlag, 1985, ch. 4, pp.313–320, 2013.

[20] H. Wang, X. Li, J. Gao, and W. Li, "MOBT: A kleptographically-secure hierarchical-deterministic wallet for multiple offline Bitcoin transactions," Future Generation Computer Systems, pp.101, 315–326, 2019.

[21] Buldas, R. Laanoja, and A. Truu, "Efficient implementation of keyless signatures with hash sequence authentication," IACR Cryptology ePrint Archive 689, 2014.

[22] Buldas, R. Laanoja, and A. Truu, "Efficient quantum-immune keyless signatures with identity," IACR Cryptology ePrint Archive 321, 2014.

[23] C. Jämthagen and M. Hell, "Blockchain-based publishing layer for the Keyless Signing Infrastructure," In 2016 Intl IEEE Conferences Ubiquitous Intelligence, and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), pp.374–381, IEEE, 2016.

[24] F. Li and P. Liu, "An efficient certificateless signature scheme from bilinear paring," In 2011 International Conference on Network Computing and Information Security (NCIS), vol.1, pp.35–37, IEEE, 2011.

[25] E. Alkim, N. Bindel, J. Buchmann, Ö. Dagdelen, E. Eaton, G. Gutoski, J. Krämer, and F. Pawlega, "FRevisiting TESLA in the quantum random oracle model," In International Workshop on Post-Quantum Cryptography, pp.143–162, 2017.

[26] C.I. Fan, Y.F. Tseng, H.P. Su, R.H. Hsu, and H. Kikuchi, "Secure hierarchical bitcoin wallet scheme against privilege escalation attacks," International Journal of Information Security, vol.19, no.3, pp.245–255, 2020.

[27] C. Yuan, M.-X. Xu, and X.-M. Si, "Research on a New Signature Scheme on Blockchain," Security and Communication Networks, 2017.

[28] G. Ra, D. Seo, M.Z.A. Bhuiyan, and I. Lee, "An Anonymous Protocol with User Identification and Linking Capabilities for User Privacy in a Permissioned Blockchain," Electronics, vol.9, no.8, p.1183, 2020.

[29] X. Luo, X. Yang, and X. Niu, "An efficient and secure outsourcing algorithm for bilinear pairing computation," In International Conference on Emerging Internetworking, Data & Web Technologies, pp.328–339, Springer, Cham, 2017.

[30] M. Zuppelli, A. Carrega, and M. Repetto, "An effective and efficient approach to improve visibility," Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), vol.12, no.4, pp.89–108, 2021.

[31] M. El-Shrkawey, M. Alalfi, and H. Al-Mahdi, "An enhanced intrusion detection system based on multi-layer feature reduction for probe and DoS attacks," Journal of Internet Services and Information Security (JISIS), vol.11, no.4, pp.61–78, 2021.

[32] L. Caviglione, S. Wendzel, A. Mileva, and S. Vrhovec, "Guest editorial: Multidisciplinary solutions to modern cybersecurity challenges," Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), vol.12, no.4, pp.1–3, 2021.

[33] Ž. Kodrič, S. Vrhovec, and L. Jelovčan, "Securing edge-enabled smart healthcare systems with blockchain: A systematic literature reviews," Journal of Internet Services and Information Security (JISIS), vol.11, no.4, pp.19–32, 2021.

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.

Su-Hyun Kim is currently a Manager in the ICT Industry Strategy at National IT Industry Promotion Agency (NIPA), Jincheon, South Korea. Recently, he has been actively working in the area of NPU (Neural Processing Unit), in particular with respect to government policies. His research interest includes cryptographic protocol, IoT security, AI security. He received Ph.D. degree in Computer Science from Soonchunhyang University, Asan, South Korea in 2016. He received M.S. degree in Computer Science from the Soonchunhyang University, Asan, South Korea in 2012 and B.S. degree in Computer Science from oonchunhyang University, Asan, South Korea in 2010.

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.