Can We Create a Cross-Domain Federated Identity for the Industrial Internet of Things without Google?

Eunsoo Kim, Young-Seob Cho, Bedeuro Kim, Woojoong Ji, Seok-hyun Kim, Simon S. Woo, and Hyoungshick Kim

Abstract

Providing a cross-domain federated identity is essential for next-generation Internet services because information about user identity should be seamlessly exchanged across different domains for authentication and authorization. Federated identity can enable users to use various services through a single account. However, conventional federated identity management systems necessarily require a *trustworthy* identity provider who stores user identity information and presents it to other service providers. Unfortunately, this requirement may not be acceptable in Industrial Internet of Things (IIoT) applications, which often require interacting and authenticating with users and devices across different domains. Who will take full responsibility for managing and issuing all digital identities for IIoT devices? Can we really trust one superpower organization to manage all the identities and credentials of IIoT devices? In this article, we provide an overview of centralized and decentralized identity management methods and examine the feasibility of those methods for IIoT applications. To overcome the inherent limitations of existing approaches, we are specifically interested in designing *decentralized* cross-domain federated identity management using *blockchain*. Our Copernican idea brings new and important perspectives in establishing universal cosmopolitan cross-domain federated identity management in a secure and fair manner.

Introduction

Maintaining multiple login identities (IDs) and credentials is always one of the most challenging problems in our world because people have difficulty in managing multiple accounts and remembering their corresponding credentials.

To address this challenge, the federated ID management [1] has been proposed and deployed as a means of linking one's electronic identities and attributes across multiple domains (e.g., organizations, corporations, applications). Federated ID management (FIM) [2] is envisioned to allow users of a certain domain to access resources from other domains seamlessly.

In a conventional FIM system, a user's credential is stored and managed at a *centralized* home organization (called "identity provider"). When a user attempts to log in to a certain application, the application sends a request to the identity provider that is the single point of reference for all parties in this FIM system. Once a user's identity is successfully verified, an identity provider responds with a security token that grants the user access to the application. For example, one of the popular approaches is the single sign-on (SSO) [3], allowing a user who has authenticated herself with a Google or Facebook account to use other applications without signing in with each application individually. This means that Google or Facebook acts as a centralized identity provider (IdPs) of applications supporting an SSO mechanism. Such centralized IdPs become attractive targets for attackers who are interested in stealing user identities. Recent cybersecurity incidents (e.g., Equifax data breach [4], Facebook security breach [5]) demonstrate that a centralized mechanism can be a major target for attacks, becoming a single point of failure.

To overcome the weaknesses of centralized approaches, decentralized blockchain-based ID management mechanisms (e.g., [6–8]) have emerged as an alternative paradigm to provide federated IDs using a blockchain, which is a decentralized transaction and data management technology across a network of untrusted parties. In this article, we analyze the pros and cons of various centralized and decentralized FIM systems and propose a specific blockchain-based ID management architecture for Industrial Internet of Things (IIoT) applications. We also provide insights on how our architecture can effectively ease and address challenges conventional FIM systems fail to solve. As IIoT applications have received considerable attention in recent years, we believe deploying a cross-domain FIM system would also become important to provide seamless communication between various IIoT devices across different domains in a secure and fair manner.

Digital Object Identifier: 10.1109/IOTM.0001.2000050

Existing Centralized ID Management Systems

We first review existing centralized approaches to provide a federated identity. SSO [3] was introduced as a solution that uses a single authentication process to permit authorized users to access all related but independent websites or applications without being prompted to sign in again at each service during a particular session. Among many proposed SSO systems, we discuss the following two most popular SSO approaches: Open Authorization (OAuth) and OpenID Connect (OIDC).

OAuth is a standardized protocol [9] that allows secure authorization in a simple and standardized way from third-party applications accessing online services, using the representational state transfer (REST) web architecture. The overall architecture of OAuth is shown in Fig. 1.

As shown in Fig. 1, a user requests services from the relying party (RP) in OAuth, where the RP is an application that wants to verify the identity of the end user. If the user is not authenticated in the RP, the RP requests the user for the user's IdP. The RP then identifies the IdP through the user identifier and requests the IdP to authenticate the user through the user's browser. The IdP authenticates the user and forwards the user's credential to the RP via the web browser. Finally, the RP verifies the user's credential for user authentication and decides whether or not to provide the service to the user. However, this approach still requires a centralized organization and authority to manage and provide services.

OIDC [10] is another SSO protocol built on the OAuth protocol to provide standard interfaces for user authentication with additional optional features, such as identity provider discovery, RP registration, signing, and encryption of messages. However, the key difference between OIDC and OAuth is that OIDC is developed for authentication only, while OAuth can also be used for authorization. OIDC handles the user's authentication process via the IdP. That is, the RP using OIDC delegates

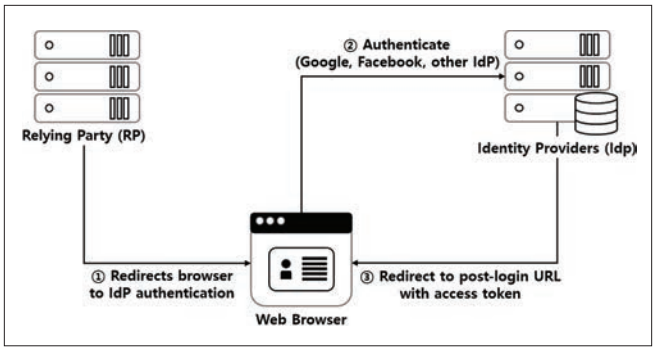

FIGURE 1. Overall architecture of OAuth.

authentication to IdP. Unlike OIDC, the primary purpose of OAuth is to ensure that the user has the proper permissions to invoke specific application programming interfaces (APIs). For example, a user can have the ability to invoke the API to fetch a list of friends.

Both OAuth and OIDC require a *trustworthy* and *reliable* central identity provider for authenticating end users. However, in IIoT applications, it is unclear who will take such responsibility for universally managing all things across multiple domains (e.g., organizations, corporations, applications) [11]. To address this problem, we believe that decentralized ID management is more suitable than centralized ID management in IIoT applications.

IIoT Applications Requiring Decentralized ID Management

To explain the necessity of decentralized ID management, we discuss the following three scenarios:

1. Drone delivery service
2. Electric car charging system
3. Peer-to-peer (P2P) energy trading network

The first use case is related to the drone delivery scenario, which will become quite popular soon. Suppose several package delivery companies (e.g., Amazon, DHL, and FedEx) use drones. In this situation, because each drone should avoid collisions with other drones during flight, it needs to interact and share its position information with other drones in real time. From the perspective of cybersecurity, however, faked drone information can be generated and exchanged between drones. Therefore, each drone should check the authenticity of other drones that communicate with it. FIM would be a neat solution for drones to communicate with each other securely. However, it would be infeasible for a single company or organization to manage all drones' identities and credentials. For example, the Chinese government may not want to share the specific flight information about Chinese companies' drones (e.g., schedule, flight route, flight speed) with a company in the United States, while it needs to broadcast minimal ID information to avoid possible collisions. To address this challenging issue, we can alternatively consider using a decentralized drone ID management scheme, which shares only the required minimal information (e.g., proof of authorized drone) among nearby drones. This scheme is necessary not only for collision avoidance but also for minimizing information leakage from each drone's flight. Moreover, the use of a decentralized drone ID management scheme can also be helpful to avoid the risk of a single point of failure that can occur in centralized FIM systems.

We consider the next use case, where companies in different industries (e.g., parking, food, energy) need to interact with one another and transfer user/device information. For example, a user charges her electric vehicle using company A to get mileage points that can be transferred to company B for free parking. Furthermore, her electric car charging bill can be recorded toward her home energy network bill managed by

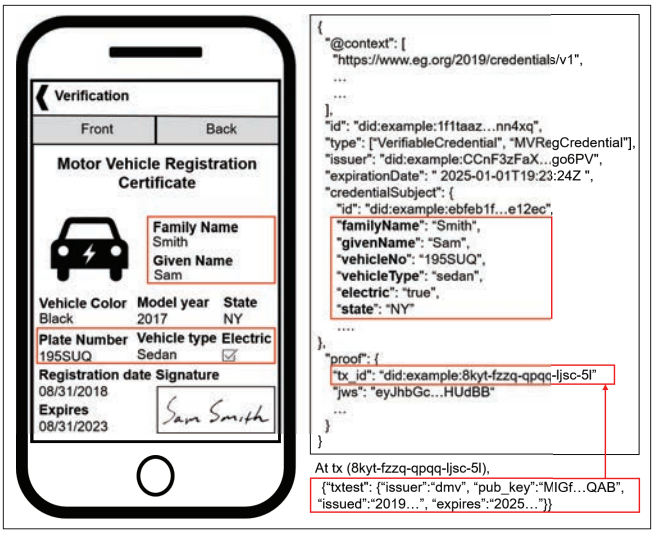

FIGURE 2. Example of the VC containing the vehicle registration information.

different company, C. Today, we have similar mileage and point systems that are being centrally managed, but they are limited to a small number of participating companies in a consortium. Consequently, this idea would not be acceptable in most IIoT applications, where a massive number of users and devices are interconnected across multiple domains (e.g., countries, organizations, corporations, applications). Without building a *universally trustworthy* ID management system that all things can trust, it is not possible to interact with (untrusted) other things or service providers across various domains at Internet scale. Decentralized ID management systems can be a promising platform for IIoT applications to identify users and things across different domains without violating anyone's right to privacy.

Lastly, we also consider the P2P energy trading scenario, where participating entities such as sellers, buyers, and aggregators need a fair and transparent mechanism to trade energy. Since trades are mainly conducted with unseen entities in a P2P fashion, it is essential to establish the trust relationships between sellers, buyers, and aggregators without a centralized trusted third party. If we assume that there is a single centralized intermediary, it seems to contradict the definition of P2P energy trading. In such a case, a decentralized ID management system is necessary to allow each node in an energy trading network to identify and authenticate other nodes that are participating in trades.

Key Requirements in Building the Decentralized ID Management for IIoT Applications

In IIoT applications, we need to provide an FIM service that allows IIoT devices from different silos such as organizations, corporations, and services to verify other devices seamlessly. Since each silo has its own trust model, decentralized ID management frameworks for IIoT applications need to establish a trust relationship among different trust models. From the scenarios mentioned above, we summarize key design requirements that must be considered when designing an FIM service for IIoT applications:

- **Neutrality:** If a company monopolizes an FIM service, there can be an issue with neutrality in managing identities between different domains in IIoT applications. For example, suppose that a customer uses company A's delivery service based on a Google-managed SSO authentication service. In this case, Google can obtain the information about the customer and shipping deal, and then Google may offer a more attractive shipping deal for the customer than company A. In this situation, Google can have a more competitive advantage over its rival companies. Therefore, neutrality should be considered for running an FIM service for IIoT applications.

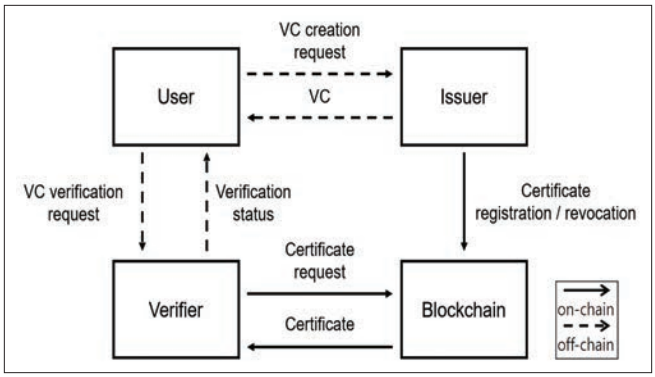

FIGURE 3. Overall information flow of the proposed system.

- **Single point of failure:** The management of identity service by a single company would be a single point of failure. Even though a company owns multiple data centers across several geographic locations, its infrastructure could still be ineffective since those data centers can be commonly vulnerable to the same type of vulnerabilities. Moreover, there are no explicit trusted third parties across heterogeneous domains, which everyone can rely on in IIoT applications.
- **Privacy:** In most existing (centralized) SSO systems, users' login and service transactions can be tracked by the IdP [12]. This means the IdP always knows at which RP the user logs in, and hence which services the user uses. In IIoT applications, devices' service usage behaviors would be sensitive and private. Therefore, an FIM service should allow users to protect and control their own data.
- **Scalability:** Since various devices should communicate with each other requiring authentication in a typical IIoT application, an FIM service for IIoT application should be scalable to be able to cope with the increasing numbers of devices. That is, an FIM service needs to concurrently process a massive number of service requests from any number of devices.

Existing Decentralized ID Management Systems

The use of blockchain can be a possible solution for decentralized ID management because transactions in blockchain are managed by a set of validators without a centralized server. Therefore, several projects were launched to offer a self-sovereign identity (SSI) that enables individuals and organizations to assert their own identity using blockchain. We introduce the following four projects for SSI using blockchain.

Sovrin [6] is the first public-permissioned blockchain to support SSI using verifiable credential (VC) [13]. VC is a technical term to represent a digitally signed credential containing a set of one or more verifiable claims made by the same entity. Figure 2 shows an example of VC, indicating that Sam Smith is the legitimate car owner with a cryptographic proof. Sovrin offers the tools and libraries as open source to create private and secure data management solutions that run on a blockchain to share digital identities. The goal of Sovrin is for users to fully manage their digital identities online. Sovrin uses standard decentralized identifiers (DIDs) [14] for identity holders to store public keys and multiple metadata.

uPort [7] is another decentralized identity platform that focuses on SSI which enables users to manage their identity using the uPort application. It is developed using the Ethereum-based decentralized public key infrastructure (DPKI) and the off-chain Inter-Planetary File System (IPFS), which is a decentralized database built on top of MongoDB to avoid a single point of failure.

Civic [8] is a blockchain platform that offers an ecosystem to provide a secure and efficient identity verification service. The goal of Civic is for users to fully manage their identities and minimize the possibility of information leakage. Users store their identification data to the Civic blockchain in an encrypted form, and the encryption key is securely protected on the user's device with his/her biometric authentication.

Despite the benefits of self-sovereign identity for both individuals and enterprises, there are still some concerns for personal data stored on blockchain. Most existing DID platforms are mainly designed to store user-related information to some extent, even though they are also trying to protect privacy. For example, in Sovrin [6], individuals' public keys should be stored in the blockchain. Similarly, in uPort [7], hash of individuals' public keys should be stored in the blockchain. Such pseudonymization techniques would be helpful to individuals' privacy, but they have potential privacy risks by linking pseudo identifiers' (e.g., public key or hash) transactions. Biryukov *et al.* [15] demonstrate that transactions in Bitcoin and Zcash can be deanonymized with high accuracy. Therefore, it is still controversial whether we can store individuals' public key or hash of identifiers, although their personal information is not directly associated [16]. To address this issue, our system is designed to avoid storing any personally identifiable information on the blockchain, including their public keys and VCs.

Furthermore, uPort and Sovrin do not provide the detailed procedure of checking the authenticity of individuals' SSI and VC upon registering them on the blockchain. In the real world, however, we cannot employ a decentralized blockchain-based ID management mechanism without considering this issue. Therefore, we alternatively focused on issuers who are responsible for issuing individuals' VC to fully design the working platform. Lastly, although many projects [6–8] have been introduced and launched for DID management, those projects are not optimized for IIoT applications and currently lack design details to develop practical systems. Unlike existing projects, we specifically focus on designing a DID management system using blockchain for IIoT applications.

Proposed ID Management Architecture

In this section, we present our DID management architecture using blockchain for IIoT applications. We aim to design a DID management architecture satisfying the four requirements we previously mentioned.

Here, we assume that a blockchain system is publicly accessible, immutable, and verifiable. Hence, anyone can access and use the issuers' certificates in the blockchain to verify the validity of individuals' VCs. In order to further minimize the burden of users (e.g., IoT devices) and system overheads in IIoT applications for interacting with a blockchain network and managing digital certificates across different domains, we only store issuers' digital certificates rather than all users' certificates and/or VCs in a blockchain. The idea of using blockchain as a certificate directory was presented in Lasla *et al.*'s work [17]. We extend their work to a more general ID management architecture by incorporating VC management schemes. Unlike their idea of using blockchain where all users' certificates are stored, we store issuers' certificates only to protect user privacy and allow a massive number of users to use the authentication service concurrently.

Entities

We first define the main entities of the proposed system, as shown in Table 1.

First, a user is a holder of a VC representing his/her/its capability and status (e.g., authorized drone). Users' VCs are digitally signed by an issuer after checking the validity of those VCs with the users' actual identity information and the issuers' databases. We use a blockchain system to securely store issuers' digital certificates (and revoked certificates). When a verifier wants to check whether a user holds a claim (e.g., "*This device is an authorized drone*"), the user delivers a VC containing the claim to the verifier who then checks the validity of the VC by verifying its digital signature with the issuer's public key retrieved from the blockchain system. Figure 3 shows how these entities

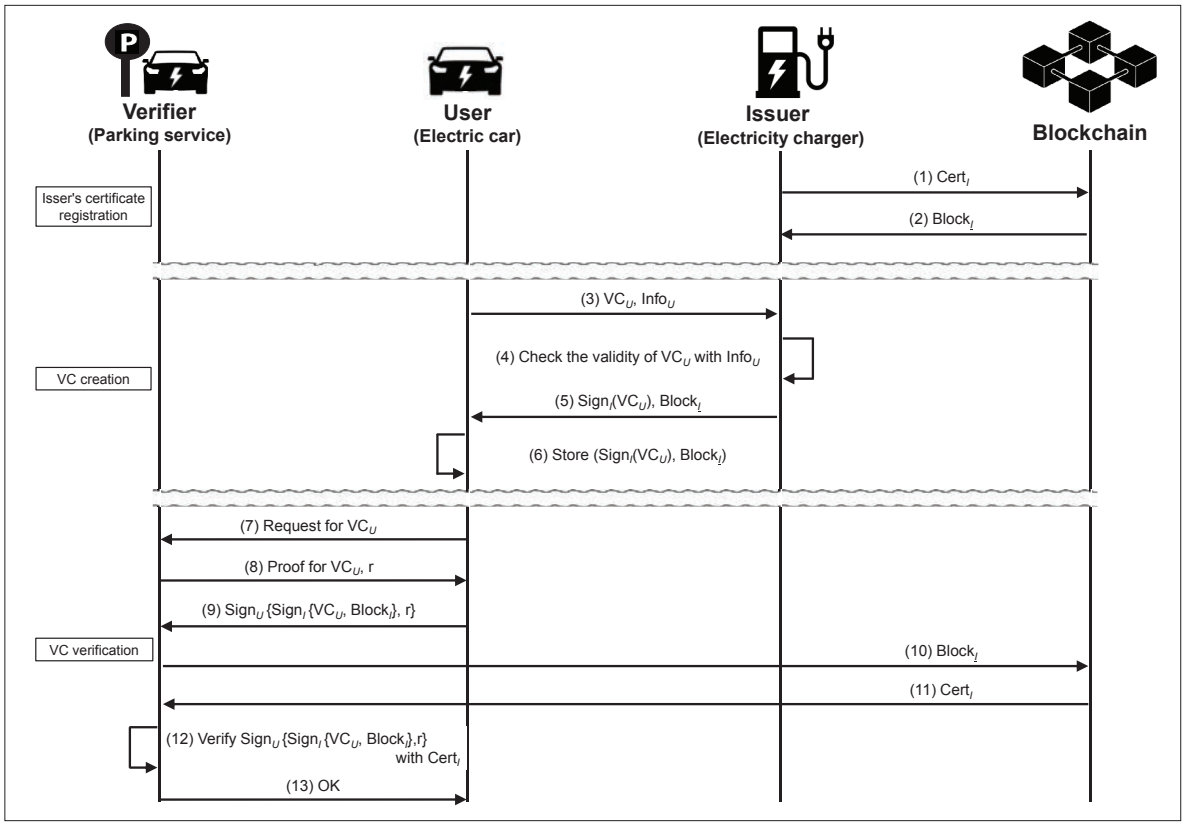

FIGURE 4. Protocol interactions among different entities in the proposed system.

communicate with each other. In this figure, we also clarify which communication protocols are on-chain (solid lines) or off-chain (dashed lines), in which on-chain transactions are the transactions available on the blockchain and visible to all the nodes on the blockchain network, while off-chain transactions refer to the transactions outside of the blockchain.

PROTOCOLS

In the proposed architecture, we have the following three procedures:

1. Issuer's certificate registration/revocation
2. VC creation
3. VC verification

Issuer's certificate registration: As shown in Fig. 4, an issuer *I* first needs to store its own certificate in the blockchain. To achieve this, issuer *I* broadcasts the certificate throughout the blockchain network, in which each block validator (or miner) has to pay some security deposit to obtain the right to create new blocks (step 1, Fig. 4). A block producer is selected among all block validators to generate a block containing transactions (i.e., registration of new digital certificates and revocation of previous digital certificates on the blockchain) that have not yet been processed from issuers. The block producer collects those transactions transmitted by issuers, and creates a new block to store them.

Next, the block is accepted as a valid block, only after verifying that the digital certificates in the block are correctly formatted and configured, and the public key in each certificate belongs to the legitimate issuer identified in the certificate. However, it is very challenging to verify the latter condition, "certificate validation process," without managing the authenticity of issuers. We discuss this issue in the next section. During the process of creating a new block, each block validator can try to check the validity of the distributed block. If the block producer creates an invalid block, the block producer loses its own deposit. Therefore, the block producer would carefully check the validity of transactions before putting them into a new block. After successfully completing the certificate registration procedure, the blockchain network sends the transaction completion message containing the block number of where the issuer's digital certificate is stored to the issuer *I* (step 2, Fig. 4).

Entity	Definition
User	A subject (e.g., person, device) who holds one or more VCs uses a VC to prove that the subject is a legitimate holder of the VC.
Issuer	An entity (e.g., government agency, network operator, blockchain node) who checks whether a user is entitled to create VCs with the user's actual information (e.g., social security number, device serial number) and digitally signs and issues a VC for a user.
Verifier	An entity (e.g., person, device, service provider) who checks the validity of VCs received from users by verifying the digital signature of the received VCs with the issuer's verification (or public) key obtained from blockchain.
Blockchain	A peer-to-peer distributed ledger that records and manages issuers' digital certificates.

TABLE 1. Main entities in the proposed system.

Issuer's certificate revocation: On the other hand, issuers sometimes need to revoke their issued digital certificates due to compromise of the corresponding private key, change of user information, or the failure of device. Therefore, we also need to perform a certificate revocation procedure when a certificate is revoked, the revoked certificate cannot be used anymore afterward, and a new certificate is added to the blockchain. In the proposed architecture, this procedure is almost the same as the certificate registration procedure shown in Fig. 4, except that the block number of the block containing the previous digital certificate is additionally delivered in step 1, Fig. 4 to revoke the previous certificate in that block. As a result of this procedure, the previous digital certificate is revoked, and a new digital certificate is added to the blockchain.

VC creation: The procedure of VC creation is depicted in Fig. 4. A user U (e.g., an electric car in our scenario) creates VC_U containing the user's desirable claims (e.g., "*XX amount of electricity was charged at an electricity charger which is a member of Alliance A*") that can be verified later according to user U's request. When VC_U is created, user U's sensitive information (e.g., credit card number, device's serial number, location) can be intentionally redacted in VC_U to improve user privacy. Next, user U sends VC_U and his/her actual information $Info_U$ (e.g., the receipt for charging) that can be used as a proof for that claim (step 3, Fig. 4). After receiving VC_U and $Info_U$ from user U, issuer I checks the validity of the claims in VC_U with $Info_U$ (step 4, Fig. 4). If issuer I believes that the claims in VC_U are valid, issuer I cryptographically signs VC_U and $Block_I$ together with issuer I's private key, and sends it back to user U where $Block_I$ indicates the block storing issuer I's certificate (step 5, Fig. 4).[1] Finally, user U stores $Sign_I\{VC_U, Block_I\}$ in its local storage to present it later for verification of using a service (step 6, Fig. 4).

VC verification: Suppose a certain amount of electricity that was charged at a member of Alliance A can be converted into mileage points for a free parking service provided by another member of Alliance A. To perform this point transfer task (Fig. 4), user U makes a request using VC_U and sends it to the verifier (e.g., a parking service company in Alliance A) (step 7, Fig. 4). After receiving the request message, the verifier asks user U to prove that U is a legitimate holder of VC_U. To achieve this objective, the verifier sends a random challenge r to user U to prevent replay attacks (step 8, Fig. 4). Upon receiving the request for a proof with r, user U computes the signature of $Sign_I\{VC_U, Block_I\}$ with r together with user U's private key corresponding to the public key in VC_U (i.e., $Sign_U\{Sign_I, VC_U, Block_I\}, r$), and sends it back to the verifier (step 9, Fig. 4). Upon receiving the signature, the verifier retrieves issuer I's digital certificate ($Cert_I$) with $Block_I$ from the blockchain database (steps 10 and 11, Fig. 4) and then verifies $Sign_U\{Sign_I\{VC_U, Block_I\}, r$ with user U's public key in VC_U and issuer I's public key in $Cert_I$ (step 12, Fig. 4). Before using $Cert_I$, the verifier also needs to check whether $Cert_I$ is revoked or not. After successfully verifying $Sign_U\{Sign_I\{VC_U, Block_I\}, r\}$ in step (12) in Fig. 4, the verifier finally sends the "OK" message to inform the user U that the user's request is accepted (see step (13) in Fig. 4).

How to Meet the Key Requirements

As mentioned earlier, the proposed architecture was designed to satisfy the four key design requirements.

To satisfy both neutrality and single point of failure requirements, we adopt a blockchain system as a certificate directory, where all issuers' certificates are stored. Any entity can register and revoke its certificate in a fair and secure manner according to the proposed protocols without relying on a trusted third party. The certificate directory promises to establish seamless links between IIoT users because multiple trust models for different domains can coexist on the proposed blockchain platform. The distributed nature of blockchain can render the proposed architecture more robust to the single points of failure in the conventional FIM [18].

To satisfy the privacy requirement, we store only issuers' certificates in the blockchain system rather than users' certificates, VCs, and other personal information. The proposed architecture was designed to reveal the minimum information about users' credential data and login transactions to protect user privacy. As shown in Fig. 3, the issuers' certificates are only exposed to the public through the blockchain network, while all users' login and service request transactions are privately performed off-chain, and VCs are stored in users' local storage.

Moreover, our design is highly scalable in that a massive number of IIoT users can concurrently use the FIM service. In a typical blockchain system, write operations are slow and expensive, while read operations can be executed concurrently with a copy of blockchain ledger in each node's local storage. In our design, write operations are only performed upon updating issuers' certificates on the blockchain by issuers' certificate registration and revocation protocols, whereas all other protocols such as VC creation and VC verification can be processed without write operations. That is, the computational and communication costs of blockchain transactions would be proportional to the number of issuers instead of the number of users in IIoT applications.

Considerations for Deployment of the Proposed System

To deploy the proposed system in the real world, we need to integrate the four entities (verifier, user, issuer, and blockchain) supporting the proposed system's protocols. VC creation and verification services can be implemented by updating conventional servers' (i.e., issuers' and verifiers') and clients' (i.e., users') programs. VCs should locally be stored at a user's device with a small storage overhead because a VC is represented as a text-based data file, as shown in Fig. 2. Probably, the most expensive infrastructural cost is to integrate a blockchain network, which can store digital certificates securely and fairly. Therefore, the proposed system's infrastructural cost can be considerably changed depending on how the base blockchain network is implemented. The simplest solution would be to use an existing blockchain system such as Ethereum and EOS, because the proposed architecture could be implemented as a decentralized application (DApp) built on smart contracts on such a blockchain system. In this case, however, we should pay to perform the operations in those smart contracts. Therefore, the DApp-based implementation results in a high cost at the issuer's side. To reduce the issuer's burden, we can alternatively develop our own proprietary blockchain platform. In this case, a consortium-based blockchain across multiple organizations seems more recommendable compared to a public blockchain, because it would be challenging to recruit volunteers who serve as block producers in the public blockchain. In the proposed system, the number of blockchain transactions would be inherently small, since new transactions are only created upon registering (or revoking) issuers' digital certificates on (or from) the blockchain. Also, typical issuers may register or revoke their certificates only a few times per year. In general, block producers' incentives would be determined by the number of blockchain transactions in public blockchains — a small number of transactions would lead to low-income expectations for block producers. Hence, the proposed architecture using public blockchain would not be attractive for block producers. Instead, we believe a consortium-based blockchain platform appears to be more promising in real-world IIoT applications.

Another challenging issue is to provide a method for checking whether a certificate on the blockchain is securely bound to a real-world issuer during the "issuer's certificate registration" procedure. Without establishing a root of trust for issuers' certificates on the blockchain system, we cannot trust any VCs signed by an issuer. To address this issue, a practical strategy we suggest is to allow verified issuers only to add their certificates to the blockchain. Our conversations with several service providers (who are interested in DID services) revealed a serious concern regarding the trust levels of issuers who can create signed VCs because malicious attackers can trick victims into believing that attackers' certificates have been created by a genuine issuer; moreover, a large number of (invalid) certificates can be created and then exchanged in order to exhaust the storage of block producers. To address this issue, again, the most promising method would be to use a consortium-based blockchain consisting of a set of trustworthy members (e.g., service providers themselves) only. As consortium-based blockchains inherently provide a membership control mechanism for blockchain users, we can use this mechanism to check issuers' authenticity and detect issuers who misbehave or do not comply with our system policies. For example, each issuer should

prove that she is a consortium member by providing her credential as a proof of membership upon registering her certificate on the blockchain. In this case, however, some advantages of the decentralized ID management scheme may be diminished because a credential management system for consortium members is needed. Another practical implementation technique is to add the URL of the issuer's official website in the issuer's certificate and publish the hash of the digital certificate via the URL. In this case, validators in the blockchain network can check whether a registered certificate belongs to the actual issuer presented in the certificate.

Conclusion

To overcome the limitations of conventional ID management systems across multiple domains, we propose a decentralized ID management system using blockchain to provide a cross-domain federated ID for IIoT applications. The proposed system would engender greater trust between things at Internet scale even when they meet each other for the first time.

Now it's time to answer our question: Can we create a cross-domain federated ID without a trustworthy party such as Google or Facebook? We might build it if we trust a blockchain network instead of a special trusted third party. We believe decentralized FIM schemes using blockchain would be a promising alternative for the IIoT world.

Acknowledgments

This work was supported by an Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government (No. 2018-0-01369) and the Information Technology Research Center (ITRC) support program (IITP-2020-2015-0-00403). The corresponding authors are Simon S. Woo and Hyoungshick Kim.

References

[1] E. Maler and D. Reed, "The Venn of Identity: Options and Issues in Federated Identity Management," *IEEE Security & Privacy*, vol. 6, no. 2, 2008, pp. 16–23.
[2] S. S. Shim, G. Bhalla, and V. Pendyala, "Federated Identity Management," *Computer*, vol. 38, no. 12, 2005, pp. 120–22.
[3] J. De Clercq, "Single Sign-On Architectures," *Proc. Int'l. Conf. Infrastructure Security*, 2002, pp. 40–58.
[4] H. Berghel, "Equifax and the Latest Round of Identity Theft Roulette," *Computer*, vol. 50, no. 12, 2017, pp. 72–76.
[5] M. Isaac and S. Frenkel, "Facebook Security Breach Exposes Accounts of 50 Million Users," Sept. 2018; https://www.nytimes.com/2018/09/28/technology/facebook-hack-data-breach.html, accessed Apr. 13, 2019.
[6] A. Tobin and D. Reed, "The Inevitable Rise of Self-Sovereign Identity," Sept. 2016; https://sovrin.org/wp-content/uploads/2017/06/The-Inevitable-Rise-of-Self-Sovereign-Identity.pdf}, accessed Apr. 13, 2019.
[7] C. Lundkvist *et al.*, "UPort: A Platform for Self-Sovereign Identity," Oct. 2016; https://whitepaper.uport.me/uPort_whitepaper_DRAFT20170221.pdf, accessed Apr. 13, 2019.
[8] J. Kilroe, "Civic: Token Behavior Model," May. 2018; https://www.civic.com/wp-content/uploads/2018/05/Token-Behavior-Model-May-16-2018.pdf, accessed Apr. 13, 2019.
[9] D. Hardt, "The OAuth 2.0 Authorization Framework," Oct. 2012; https://rfc-editor.org/rfc/rfc6749.txt, accessed Apr. 13, 2019.
[10] N. Sakimura *et al.*, "OpenID Connect Core 1.0," Feb. 2014; https://bitcoin.org/bitcoin.pdf, accessed Apr. 13, 2019.
[11] F. Toesland, "Top 5 Applications for the Industrial Internet of Things," Mar. 2017; https://www.raconteur.net/technology/top-5-applications-for-the-industrial-internet-of-things}, accessed Apr. 13, 2019.
[12] D. Fett, R. Küsters, and G. Schmitz, "SPRESSO: A Secure, Privacy-Respecting Single Sign-On System for the Web," *Proc. 22nd ACM SIGSAC Conf. Computer and Commun. Security*, 2015, pp. 1358–1369.
[13] M. Sporny, D. Longley, and D. Chadwick, "Verifiable Credentials Data Model 1.0: Expressing Verifiable Information on the Web," Nov. 2019; https://www.w3.org/TR/verifiable-claims-data-model/}, accessed Mar. 17, 2020.
[14] D. Reed *et al.*, "Decentralized Identifiers (DIDs) v0.13: Data Model and Syntaxes," Aug. 2019; https://w3c-ccg.github.io/did-spec/, accessed Sept. 28, 2019.
[15] A. Biryukov and S. Tikhomirov, "Deanonymization and Linkability of Cryptocurrency Transactions Based on Network Analysis," *Proc. IEEE European Symp. Security and Privacy*, 2019, pp. 172–84.
[16] M. Finck, "Blockchains and Data Protection in the European Union," *European Data Protection Law Review*, vol. 4, no. 1, 2018, pp. 17–35.
[17] N. Lasla *et al.*, "Efficient Distributed Admission and Revocation Using Blockchain for Cooperative ITS," *Proc. 9th IFIP Int'l. Conf. New Technologies*, 2018, pp. 1–5.
[18] A. Yakubov *et al.*, "A Blockchain-Based PKI Management Framework," *IEEE/IFIP Network Operations and Management Symp.*, 2018, pp. 1–6.

Biographies

Eunsoo Kim (eskim86@skku.edu) is a Ph.D. student in the Department of Computer Science and Engineering, Sungkyunkwan University, Korea. He received an M.S. degree from the Department of Computer Science and Engineering, Sungkyunkwan University. His current research interest is focused on network security, cyber threat intelligence, and security engineering.

Young-Seob Cho (yscho@etri.re.kr) is a principal researcher in the Information Security Research Division of ETRI. He received a Ph.D. degree in computer science from Inha University, Korea. Since 1998, he has worked on research projects at ETRI and continuously conducted numerous projects on national and international level. His current research interests lie in blockchain identity management, AI security, authentication, and authorization.

Bedeuro Kim (kimbdr@skku.edu) is a Ph.D. student in the Department of Computer Science and Engineering, Sungkyunkwan University. He received an M.S. degree from the Department of Computer Science and Engineering, Sungkyunkwan University. His current research interest is focused on usable security and security engineering.

Woojoong Ji (woojoong@skku.edu) received his B.S. degree from the Department of Information Security, Joongbu University, in 2018. He is currently a graduate student with the Department of Computer Science and Engineering, Sungkyunkwan University, supervised by Hyoungshick Kim. His current research interests include network security, mobile security, IoT security, and security engineering.

Seok-Hyun Kim (ksh4uu@etri.re.kr) is a senior researcher in the Information Security Research Division of ETRI. He received an M.S. degree in information security from Chonnam University, Korea. Since 2010, he has worked on research projects at ETRI and continues to carry out various national projects. His current research interests lie in blockchain identity management, FIDO, authentication, and privacy.

Simon S. Woo [M] (swoo@skku.edu) received his M.S. and Ph.D. in computer Science from the University of Southern California, Los Angeles, his M.S. in electrical and computer engineering from the University of California, San Diego, and his B.S. in Electrical Engineering from University of Washington, Seattle. He was a member of technical staff (technologist) for nine years at NASA's Jet Propulsion Lab (JPL), Pasadena, California, conducting research in the satellite communications, networking, and cybersecurity areas. Also, he worked at Intel Corp. and Verisign Research Lab. Since 2017, he has been a tenure-track assistant professor at SUNY, South Korea and a research assistant professor at Stony Brook University. Now, he is a tenure-track assistant professor with the SKKU Institute for Convergence and Department of Applied Data Science and Software at Sungkyunkwan University.

Hyoungshick Kim (hyoung@skku.edu) is an associate professor in the Department of Computer Science and Engineering, Sungkyunkwan University. He is also working as distinguished visiting researcher at CSIRO Data61. He received a B.S. degree from the Department of Information Engineering at Sungkyunkwan University, an M.S. degree from the Department of Computer Science at KAIST, and a Ph.D. degree from the Computer Laboratory at the University of Cambridge in 1999, 2001, and 2012, respectively. He previously worked for Samsung Electronics as a senior engineer from 2004 to 2008. His current research interest is focused on usable security and security engineering.

Footnotes

[1] We note that the metadata about issuer I can actually be included in VC_U according to the DID standard [14]. However, in this article, we explicitly show $Block_I$ for clarification because we do not explain the detailed field information about VC_U.