Ensuring Data Security in the Context of IoT Forensics Evidence Preservation with Blockchain and Self-Sovereign Identities

Cristian Alves dos Santos(✉), Leandro Loffi, and Carla Merkle Westphall

Computer Science, Federal University of Santa Catarina (UFSC), PO Box 476, Florianopolis, SC 88040-970, Brazil
{cristian.alves,leandro.loffi}@posgrad.ufsc.br, carla.merkle.westphall@ufsc.br

Abstract. As Internet of Things (IoT) networks expand, significant challenges related to the secure management of data generated by these devices emerge. The integrity and reliability of this data are critical in sensitive sectors, such as forensic evidence preservation. In this context, we present an innovative architecture based on Self-Sovereign Identity (SSI) tailored for resource-constrained IoT devices.

Our proposal addresses the intrinsic limitations of current systems, which often fail to ensure the integrity, reliability, and traceability of data originating from IoT devices. To tackle this issue, we propose using decentralized identifiers (DIDs) to establish unique identities for IoT devices, accompanied by verifiable credentials (VCs) that attest to data ownership. To implement this solution, we have developed an application that serves as a gateway for resource-constrained devices, typically certified and connected to a broker. Our application utilizes Hyperledger Aries and Indy libraries, providing essential resources to address these challenges. Furthermore, we conducted comprehensive simulations and a performance analysis to validate the effectiveness of our approach. Integrating these technologies enables the certification of data collected by IoT devices, offering a robust framework for the data custody chain. Consequently, this substantially contributes to preserving this data's integrity, reliability, and traceability in critical environments.

Keywords: Internet of Things · Self-sovereign identity · Decentralized identifiers · Verifiable credentials

1 Introduction

With the rapid growth and continuous evolution of intelligent devices and systems, also known as the Internet of Things (IoT), new services drive complex interactions among these devices, services, and people. However, this growth and the exponential increase in the number of connected IoT devices generating and

V. Muthukkumarasamy et al. (Eds.): ICISS 2023, LNCS 14424, pp. 319–338, 2023.
https://doi.org/10.1007/978-3-031-49099-6_20

processing massive volumes of data raise significant concerns about the security of this information [1,2].

In this context, the traceability and verification of data generated by these devices play a crucial role in managing the data custody chain to preserve forensic evidence. Digital forensic investigation plays a pivotal role in virtually all criminal investigations, given the abundance of available information and the opportunities presented by electronic data to investigate and substantiate crimes. However, during legal proceedings, these electronic pieces of evidence are often met with extreme suspicion and uncertainty, although in some situations, they are justified. The use of scientifically unproven forensic techniques is widely criticized in current legal procedures. Furthermore, electronic data's highly distinct and dynamic characteristics, combined with existing legislation and privacy laws, continue to pose significant challenges to the systematic presentation of evidence in a court of law [3].

Throughout all phases of forensic investigation, digital evidence is susceptible to external influences and contact with various factors. The legal admissibility of digital evidence is the capacity for this evidence to be accepted as proof in a court of law. The probative value of digital evidence can only be preserved if it is possible to establish that the records are accurate, meaning who created them, when they were created, and that no alterations have occurred [4].

Therefore, it is imperative to establish unequivocal device identification and ensure the integrity and authenticity of the generated data. These objectives can be achieved by adopting technologies based on decentralized identifiers (DIDs) [5] for identification and using verifiable credentials to certify the authenticity of measurements sent by IoT devices. This approach is essential in instilling trust and integrity in communications and transactions, bridging a significant gap in existing systems, and providing an additional layer of security and reliability in interactions between IoT devices and the systems to which they are connected.

By adopting Self-Sovereign Identity (SSI), IoT devices can be securely and reliably identified, establishing a unique identity and ensuring that transactions and communications are conducted authentically and immutably. This approach is particularly relevant in the face of significant challenges related to the sharing and preserving of forensic data in technological, legal, and operational terms. Demonstrating data integrity is crucial to ensure its validity and admissibility in legal proceedings.

However, IoT devices often have limited resources in terms of low processing power, storage, memory, and limited battery life [1]. This imposes a limitation on the use of DIDs and renders them incapable of storing digital wallets, running an agent, or maintaining the necessary protocol stack to enable SSI capabilities. Furthermore, IoT devices require cryptographic solutions to meet security, privacy, and trust requirements, which are crucial for enabling the use of DIDs and Verifiable Credentials (VCs) [6].

This limitation can pose a challenge in adopting these advanced technologies on resource-constrained devices, necessitating alternative approaches to ensure the security and integrity of communications and transactions in this specific

context. One practical solution is to securely outsource processing to a more powerful external device to reduce the computational cost of cryptographic calculations and maintain data confidentiality.

To address these issues, we propose an SSI Gateway for IoT to identify and certify data emitted by these devices. Our approach seeks to establish means for managing the custody chain of data issued in the context of forensic evidence preservation. This architecture utilizes a blockchain-based SSI model [7] to safeguard the privacy of data collected from IoT devices.

This approach allows an IoT device, whether an emitter or connected to an emitter in a certified manner, to sign the data at its source when transmitting it, making it tamper-proof and verifiable, potentially rendering it trustworthy as long as the emitter is reliable. Consequently, the involved parties can preserve ownership of the collected information, enabling the sharing of verifiable credentials certifying the integrity and origin of data collection.

In summary, our main contributions are as follows:

1. We implemented an SSI Gateway architecture designed for IoT networks.
2. We developed a system for issuing verifiable credentials to ensure the integrity and traceability of data from IoT devices.
3. We implemented decentralized identifiers to strengthen the identification of devices within the architecture, thereby enhancing data custody chain security.

In our experiments, we utilized the Hyperledger Indy Software Development Kit (SDK) [8] to generate DIDs for device identification. We utilized Hyperledger Aries Cloud Agent Python (ACA-Py) [9] as a cloud agent for issuing VCs and establishing connections with other parties. The Von Network [10] was adopted as a permissioned blockchain to anchor DIDs, credential metadata, and verification keys. To assess the performance of the proposed architecture, we conducted simulations and, through performance testing during the DIDs registration and VCs issuance process, analyzed the results and identified potential areas for optimization.

The remaining part of the article is organized as follows: In Sect. 2, we provide the essential background. Next, Sect. 3 discusses related works and the techniques they address. In Sect. 4, we present an overview of the proposed architecture. Subsequently, Sect. 5 delves into the detailed performance test results and analyzes these findings. Finally, in Sect. 6, we discuss the conclusions drawn from this study and explore potential future research directions.

2 Background

2.1 Self-Sovereign Identities

Self-sovereign identity is a concept that describes an individual's ability to control and manage their digital identity in a decentralized manner. [11] first proposed the idea of self-sovereign identity and has been referenced by various

authors in the literature. While there still needs to be a consensus on SSI, the widely accepted concept is described as a system where individuals can claim and manage their identity without needing a centralized trusted party [12].

SSI allows users to choose which identity information to share and with whom, as well as granting them access to their identity information at any time. Instead of relying on third parties, such as companies or governments, to manage and store their identity information, users can create, store, and control their digital identities through decentralized technologies like blockchain.

Decentralized Identifiers. Decentralized Identifiers are globally unique identifiers that enable an entity's verifiable and persistent identification for as long as the DID controller desires without relying on a centralized registry [13]. Just as there are various types of URIs complying with the URI standard, there is also a variety of DID methods, all of which must adhere to the DID standard [5]. Each DID method specification should define the name of the DID method, which should appear between the first and second colons (e.g., did:method:unique-identifier), the structure of the unique identifier following the second colon, and the technical specifications outlining how a DID resolver can implement operations to create, read, update, and deactivate a DID document using that method.

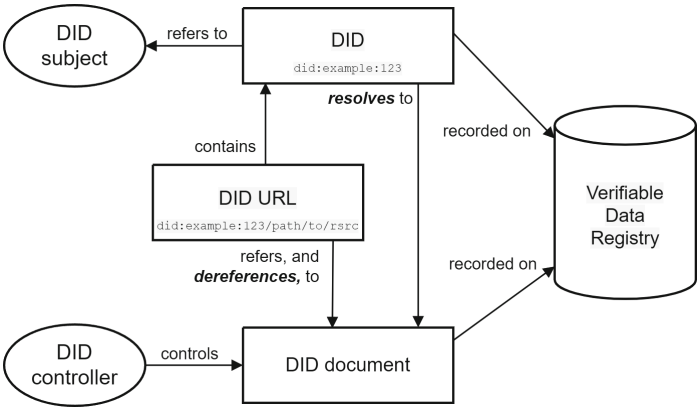

Fig. 1. Overview of DID architecture and the relationship of the essential components.

Figure 1 depicts a conceptual diagram of the W3C-proposed [5] DID architecture. A DID is linked to a DID document containing relevant DID-related information. This DID document can be accessed by resolving the DID itself. Within this document, essential elements for authentication and verification, including the DID itself and associated public key information, can be found. Hence, the DID document is pivotal in conducting secure authentication and verification processes using DID.

A DID points to a DID document, a JavaScript Object Notation (JSON) file with a format defined explicitly in the DID specification, and a set of information

describing the DID subject. This document includes cryptographic public keys that the DID subject and a DID delegate can use to authenticate and verify their connection to the DID.

Each entity possesses one or more DIDs, which can be resolved through a DID Resolver, much like the Domain Name System (DNS) functioning in the context of SSI. When a valid DID is provided to a DID Resolver software, it operates as a browser, receiving a Uniform Resource Locator (URL), resolving the DID, and providing a DID document in response [14].

Verifiable Credentials. A verifiable credential is a digital representation of information typically found in physical credentials but with enhanced security and reliability, thanks to digital signatures. A credential comprises claims made by an entity, a person, an institution, or a machine, and metadata such as issuer, expiration date, representative image, and public key for verification [6].

A verifiable credential is founded upon a "triangle of trust," wherein the issuer trusts the subject, the subject trusts the verifier, and the verifier trusts the issuer. Depending on the context, this trust relationship can be established concerning individuals, institutions, or machines. The process of a verifiable credential consists of essential steps:

1. Recording the DID and Public Key on the Blockchain: The issuer or entity issuing credentials writes the DID along with its public key onto a blockchain or another trusted public service. This establishes a connection between the issuer and their DID.
2. Issuance of Verifiable Credentials: The issuer uses their private key to sign a verifiable credential digitally. This credential is then issued to a qualified subject, who stores it in their digital wallet.
3. Request for Proof by the Verifier: A verifier seeking to confirm the identity or specific information of the subject requests digital proof of the credentials from the subject. If the subject agrees, they generate and send the proofs to the verifier from their digital wallet.
4. Verification of Proofs by the Verifier: The proofs contain the issuer's DID. The verifier uses this DID to retrieve the issuer's public key and other cryptographic data from the blockchain. Subsequently, the verifier utilizes the issuer's public key to ascertain the validity of the proofs and whether the digital credential has been tampered with.

In the context of verifiable credentials, the blockchain serves as a registry where issuers publish cryptographic keys and credential metadata. This enables credential holders to generate presentations that verifiers can cryptographically verify. However, it is crucial to emphasize that, to ensure data privacy and security, storing the credentials themselves on the blockchain is not recommended. Instead, the standard practice is to store only the public keys of the issuer and the holder on the blockchain linked to their respective DIDs. Credentials containing personal information are securely kept in a private digital wallet. The availability of these keys is sufficient to verify the credentials, eliminating the need to store them on the blockchain [14].

Zero Knowledge Proof. Zero-knowledge proof (ZKP) is a cryptographic technique that allows one to prove possession of certain information without revealing its content, ensuring data privacy and transaction security. The Verifiable Credentials model combines the implementations of zero-knowledge proofs and verifiable credentials to reduce data and increase privacy. With this combination, provers can present proofs without disclosing sensitive and personal data to verifiers. In order to enhance privacy and security, the claims present in a verifiable credential can be exposed as a predicate or selective disclosure of zero-knowledge proof [15].

2.2 Relationship with Blockchain Technology

Blockchain exhibits characteristics that align with the desired properties of SSI. For instance, blockchain provides a decentralized domain that is not controlled by any single entity. Data stored on any blockchain is readily available to any authorized entity. The owner of specific data has complete control over it and determines how it can be shared with other users within the blockchain domain, thus satisfying the principle of ownership and disclosure.

In this context, Hyperledger Indy is one of the most advanced technologies concerning self-sovereign identity. This platform provides robust and innovative solutions for secure and decentralized digital identity management. In Hyperledger Indy, private content is not stored on the blockchain, ensuring enhanced privacy preservation in solutions utilizing this distributed ledger technology. The blockchain maintains only the public DIDs, schemas, credential definitions, and revocation records. This approach ensures that sensitive information is not exposed to the network, making the architecture more resistant to potential attacks or future vulnerabilities [15].

2.3 Use of Agents

Agents can represent individuals, organizations, or devices in SSI ecosystems. They are software responsible for securely managing and using DIDs and VCs stored in digital identity wallets. These software entities require access to the wallet to perform cryptographic operations on behalf of the represented entity. Their responsibilities encompass various essential tasks such as message sending and receiving, information encryption and decryption, digital signature on behalf of the entity, wallet management, and backup/restoration of information. Additionally, some agents can interact with the ledger, enabling the adoption of verifiable data models [15,16]. Among the existing agents, Hyperledger Aries [17] stands out as an advanced and comprehensive solution. It implements the Hyperledger Ursa [18] cryptographic library, ensuring a high level of security in cryptographic operations.

3 Related Work

From a research perspective, only a few studies have proposed the application of the SSI paradigm to IoT. Following a search for relevant keywords such as

"Internet of Things," "Self-Sovereign Identity," "Decentralized Identifiers," "Verifiable Credentials," and "Distributed Ledger Technology" in prominent research sources, several works were identified that briefly mentioned the intersection of SSI and IoT. While these studies provide an overview, they do not delve into credential exchange protocols or technical details. However, there are standout studies known for their specific proposals:

Initially, authors [19] explored the SSI paradigm for IoT, introducing DIDs and VCs for IoT. They also analyzed SSI's application in IoT ecosystems, convincingly demonstrating that this approach surpasses traditional certificates such as Pretty Good Privacy (PGP) and X.509 regarding privacy and effectiveness.

SSI can leverage a decentralized registry to store information such as DIDs, DID documents, and verifiable credential metadata. This registry can be implemented using Distributed Ledger Technology (DLT), enhancing the system's overall security while ensuring the integrity and availability of stored information. In this regard, several studies have conducted research.

For instance, authors [20] introduced an SSI scheme based on IOTA as a DLT to implement decentralized identity. They underscored its permissionless nature, absence of transaction costs, and scalability advantages. However, a significant limitation of this proposal lies in IOTA's need for complete decentralization, as it still relies on a coordinator, a centralizing element in the consensus process.

Some authors introduced concepts related to SSI and illustrated specific use cases for industrial IoT (IIoT). [21] proposed a protocol for device identity management based on Hyperledger Indy but did not provide implementation details. [22] suggested a digital identity framework for devices using a combination of Hyperledger Indy and Ethereum. However, the authors should have comprehensively addressed scalability issues, as the Ethereum platform entails costs that could pose significant challenges when dealing with increasing demand.

Other studies delve into the SSI paradigm for Medical Internet of Things (MIoT) devices. The authors [23] conducted a study on authentication mechanisms for medical devices. However, the proposal must discuss results and address performance and scalability issues. Some studies have addressed IoT as a service. For instance, the authors [24] introduced an SSI-based identity management system for the IoT-as-a-Service (IoTaaS) business model. However, the tests were conducted on specific devices with memory and processing capacity without considering the limitations of resource-constrained devices.

Table 1 compares our proposal, and various related works in the field of SSI applied to IoT. Each row represents a different study, identifying the authors, the application domain of the work, and the addressed problem. The first four criteria are vital indicators for analyzing the solution's suitability for constrained IoT devices and data integrity. It also considers scalability and performance evaluation. Our proposal stands out comprehensively and successfully addresses all these essential aspects.

Regarding devices with limited resources, the authors [25] proposed using DIDs as identifiers for IoT devices and conducted a precise examination of the requirements for IoT devices to implement an SSI-based identity management

Table 1. Comparison between related works and our proposal.

Authors	Application Domain	Issue Addressed	Constrained IoT devices	Data Integrity Verification	Scalability	Performance Evaluation
[19]	IoT devices	Digital Identity	×	×	×	×
[20]	IoT devices	Data accessing	–	×	✓	✓
[21]	IoT Industrial	Data accessing	×	✓	×	×
[22]	IoT Industrial	Efficiency of data sharing	×	×	×	✓
[23]	IoT Medical	Access control	×	✓	×	×
[24]	IoT Services	Secure transactions	×	✓	×	✓
[25]	IoT devices	Digital Identity	✓	✓	–	×
[26]	IoT devices	Digital Identity	✓	×	–	×
[27]	IoT vehicles	Data integrity	×	✓	–	×
[28]	IoT vehicles	Data security	×	✓	–	×
This work	IoT Data	Data integrity verification	✓	✓	✓	✓

system. They also put forward a proxy-based approach. Other authors [26] have also proposed proxy-based approaches, such as IoT Exchange, to establish the connection between IoT devices and users. However, it is worth noting that this proposal does not provide a concrete specification for implementing VCs for IoT, primarily focusing on analyzing DIDs as suitable identifiers for this specific environment.

In the study by [27], a framework was introduced to verify the authenticity of vehicle emission values through a decentralized authentication and authorization system utilizing blockchain technologies (Hyperledger Fabric and Indy). However, it is essential to consider the presence of central entities, including Registration Authorities (RAs), which raise questions about the actual degree of decentralization and control within the ecosystem. On the other hand, [28] presented a model for secure software updates in the ecosystem of embedded devices in vehicles, using a decentralized architecture with Hyperledger Indy. Nevertheless, the study needs more in-depth technical implementation details.

In this regard, it is essential to note that none of these studies addresses the specific nuances related to the data custody chain, which are crucial to ensuring data integrity, validity, and traceability, especially in resource-constrained devices. Furthermore, few research efforts have been dedicated to performance evaluation in similar contexts, making it challenging to draw comparisons to enhance this critical aspect within the IoT context.

4 System Overview

The proposal we present has as its primary objective the issuance of verifiable credentials to present and substantiate claims regarding data emitted by IoT

devices. Furthermore, we propose identifying IoT devices and the traceability of the data they generate using Decentralized Identifiers anchored in a blockchain infrastructure. Both features aim to strengthen the integrity and validity of data from its source, thus contributing to a more robust and reliable chain of custody.

4.1 Definition of Tools and Technologies

We have used the Sovrin method [29] for DIDs, identifying each device and registering each data emission. Furthermore, we employ VCs in the AnonCreds [30] format to establish a robust foundation of verifiable evidence based on the data collected by IoT devices.

To implement these features, we used Hyperledger Indy and Aries. When evaluating platforms for SSI, we considered fundamental criteria such as the scalability of the permissioned blockchain and coverage of concepts grounded in SSI.

In Hyperledger Indy, unlike other DLTs, incentives are not required. As a result, all transactions encompassing various operations, such as the creation of DIDs, key rotation, credential schema creation, credential definition, and other functionalities, can exhibit improved performance [31]. Regarding the consensus algorithm, Indy employs Practical Byzantine Fault Tolerance (PBFT), enabling a high transaction rate [32]. The performance results presented in [33] demonstrate that Indy meets the criteria for global scalability in terms of record query speed.

We employ Hyperledger Aries as the agent for our Gateway, which provides capabilities such as interacting with other agents and the blockchain, supplying libraries for cryptographic wallet implementation, sending event notifications, and receiving instructions from the controller.

To facilitate communication with constrained devices, we utilize the Message Queuing Telemetry Transport (MQTT) protocol, designed for resource-limited devices [34]. For these devices, typically connected to a broker, we use Mosquitto. Both the devices and our application establish secure and certified connections to interact with messages transmitted by the broker.

These platforms and tools stand out as suitable choices for our architecture, aiming to meet the demands of the data custody chain in forensic scenarios.

4.2 Gateway-Based Approach for Constrained IoT Devices

Our approach is built upon an architecture structured into three interconnected and collaborative layers: Edge, Fog, and Cloud Computing. Figure 2 illustrates these layers along with their respective entities.

The Edge layer encompasses a variety of IoT devices, such as sensors, medical devices, surveillance cameras, and other manually embedded devices. In this context, the broker is crucial in receiving and transmitting data from these constrained devices.

The Gateway acts as an intermediary between the Edge and Cloud layers in the Fog Computing layer. At the core of the Gateway, the controller provides

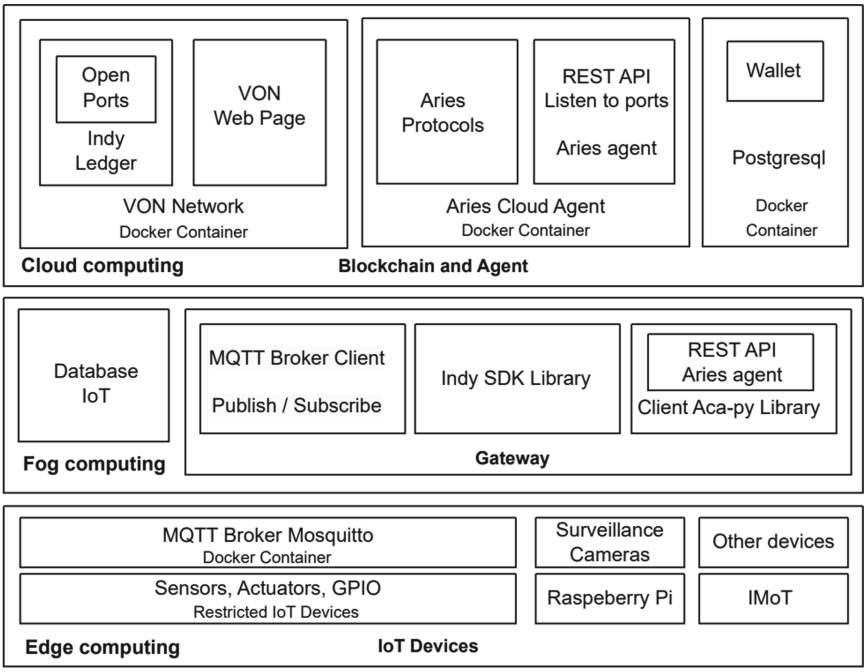

Fig. 2. Layered overview of the proposed architecture.

essential resources such as storage, processing capacity, and sources of entropy for cryptographic key generation through the functionalities of the Indy library, which also enables the implementation of DIDs and access to blockchains. Additionally, the Gateway performs the functions of a cloud agent controller, establishing secure connections with other agents and implementing features related to VCs.

In the Cloud Computing layer, ACA-Py is responsible for registering on the blockchain the schemas and credential definitions previously established in the previous layer. Registering DIDs previously set on the blockchain makes it possible to verify credentials using public keys. This enables decentralized verification of digital identities associated with devices and the data they emit in our architecture.

4.3 Use Case in Evidence Preservation

In the context of a use case scenario, our proposed architecture aims to establish a chain of custody for data originating from restricted devices. This approach seeks to create an environment that ensures forensic evidence's effective and reliable preservation.

Generally, the identification of devices connected to the broker is carried out at the Gateway. A unique DID is generated using the "did: sov" method

when a new device is added to the system. This DID is subsequently registered on the blockchain with its alias: a serial number, UUID (Universally Unique Identifier), or MAC address (Media Access Control). This association establishes an immutable relationship between the device and its identification.

Optionally, when the broker receives data, the Gateway captures this data and creates a DID using the same method, which is then registered on the blockchain. This enables the identification and tracking of the collected evidence.

Subsequently, a verifiable credential is generated for the stakeholder, which can be a regulatory authority, regulatory entity, or a court. Another use case scenario involves on-demand credential issuance, with the data securely stored in a database and the credential issued upon request.

With this framework, it becomes possible to generate verifiable credentials containing information derived from IoT devices, where each attribute is given a signature. This approach ensures that the credential provides a tamper-proof and secure representation of the collected data. Interested parties can verify the authenticity and integrity of the data through a distributed ledger using the issuer's public key, enhancing trust in IoT applications and data sharing.

In the usage scenario depicted in Fig. 3, the intention is to present an architecture in which an application serves as a proxy between resource-constrained IoT devices and the cloud agent ACA-Py, which has communication capabilities with stakeholders and credential issuance abilities.

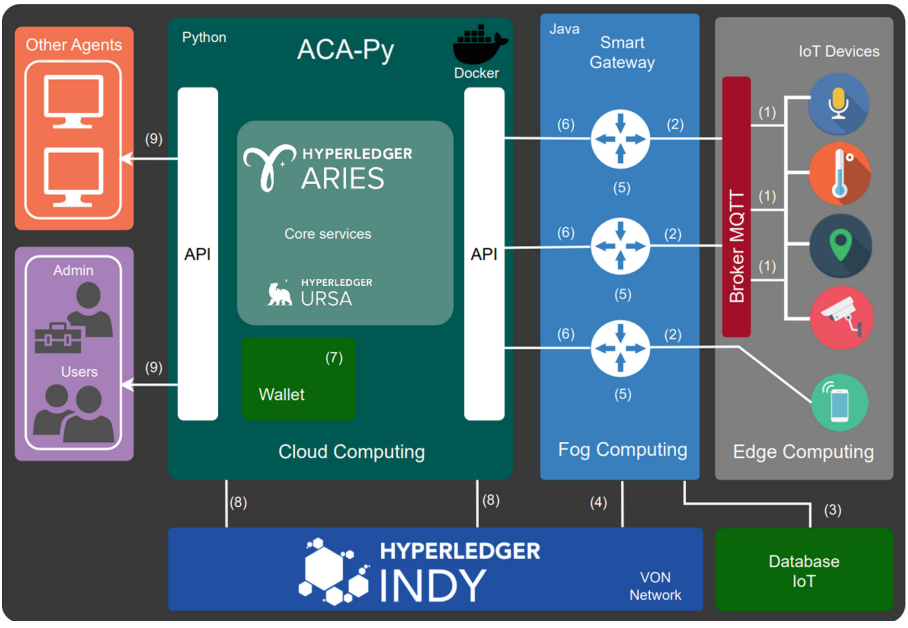

Fig. 3. Overview of the Proposed Architecture.

The following steps describe the sequence of the proposed architecture:

1. In the Edge Computing layer, devices establish secure connections using certificates and transmit data to the broker using the MQTT protocol.
2. In our application in the Fog Computing layer, we establish a secure connection with the broker using certificates and perform a prior subscription to the broker's message topic to receive data.
3. The received data is stored in an internal database to facilitate queries.
4. In the same layer, devices are registered. The DID and the device's serial number are registered once on the blockchain for identification and tracking. Additionally, another registration is conducted on the blockchain to identify each data collection associated with the corresponding device's DID.
5. After the device emits the data, a package is assembled, incorporating this information, the timestamp, device type, location, issuing entity, and collected data. In the data flow context, a generated file that cannot be directly included in the credential can be identified through a hash, verifying its integrity. Additionally, both the hash and the link to the corresponding storage location are recorded in the credential.
6. The Gateway, acting as a controller, uses Application Programming Interfaces (APIs) to access the cloud agent's resources and issue the credentials. The credential is issued to the stakeholder and stored in their wallet.
7. In this context, the Indy wallet is implemented using a PostgreSQL database. It stores cryptographic keys, DIDs, VCs, and other sensitive information necessary for interaction with the network.
8. The agent interacts with the blockchain to record the necessary transactions, known as NYM, which enable the creation of DIDs, ATTRIB, schema, and credential definitions. This interaction ensures the consistency and security of transactions within the proposed architecture, enabling the verification and validation of issued credentials.
9. Authorized users linked to the issuer can query the issued credentials. Holders can store these credentials and present them to the verifier or generate zero-knowledge proofs when compiling verifiable presentations before transmission. Interested entities can verify authenticity whenever necessary by querying the blockchain through the public key to which the credential was signed and verify integrity through the signature hash of each credential attribute. Furthermore, tracing the DID of the collection and device on the blockchain is possible, providing an additional layer of security to ensure data custody.

Thus, the Gateway enables secure and efficient communication between IoT devices and other domains. The utilization of VC resources in this context can be employed to preserve the integrity and authenticity of information.

In Fig. 4, we present a small portion of the JSON structure that comprises the issued credential based on telemetry data collected from IoT devices. This data includes the DID previously assigned to the device, the DID associated with each transmission, timestamps, device location, and the transmitted sensor readings.

```
"credential_proposal": {
  "@type": "https://didcomm.org/issue-credential/1.0/credential-preview",
  "attributes": [
    {
      "name": "Operator",
      "value": "LRG UFSC"
    },
    {
      "name": "device_did",
      "value": "did:sov:7jfceWZRA7jYdb866iFHRR"
    },
    {
      "name": "collection_did",
      "value": "did:sov:L5DknuZMuzchXyXLvmfAuX"
    },
    {
      "name": "data",
      "value": "50.5F"
    },
    {
      "name": "device_type",
      "value": "TemperatureSensor"
    },
    {
      "name": "location",
      "value": "living_room"
    },
    {
      "name": "timestamp",
      "value": "2023-10-02 00:31:03.669"
    }
  ]
},
"schema_id": "B3qPp5s37rQNYpWSUs9bXC:2:lrg:1.0",
"cred_def_id": "B3qPp5s37rQNYpWSUs9bXC:3:CL:82542:lgr"
```


Fig. 4. JSON Format Credential Proposal.

In the sequence diagram of the proposed architecture, depicted in Fig. 5, it is possible to observe the interactions among different parts of the architecture, highlighting the interactions between entities across all phases.

In this context, a mutual authentication process is conducted to establish secure communication with other agents, where both parties need to demonstrate possession of the signature keys corresponding to the paired DIDs. Following the successful completion of mutual authentication, credentials are signed using the verification keys and sent to other agents, ensuring the confidentiality of transmitted information.

4.4 Security Considerations

It is crucial to consider critical aspects related to the security and privacy of the proposed architecture.

Protecting cryptographic keys in the wallet is essential in the signing processes and data custody management. This protection can be strengthened through the use of Secure Enclaves [35].

To ensure the privacy of device attributes and collected data, one can utilize the Zero-Knowledge Proof technique when compiling verifiable presenta-

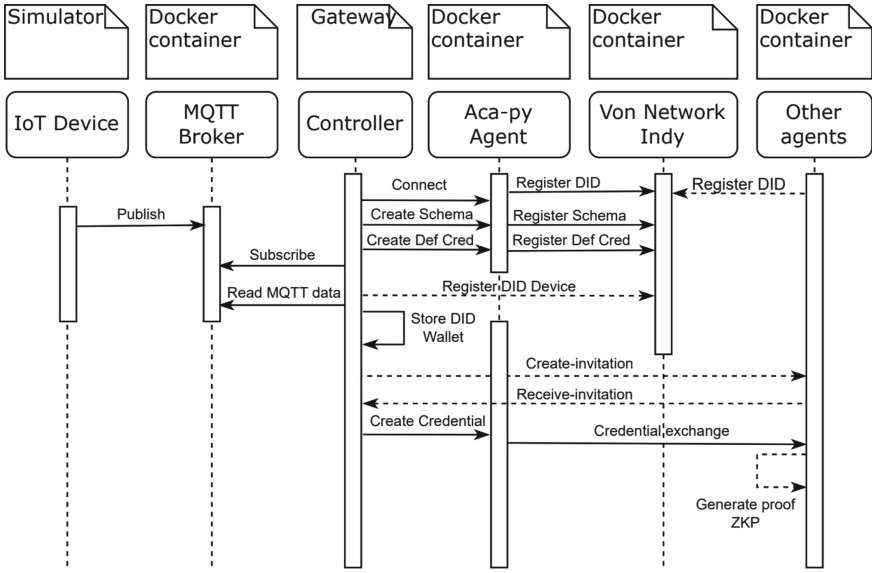

Fig. 5. Sequence diagram of the proposed architecture.

tions before sharing them with stakeholders. Another crucial aspect is aggregation, which occurs when stakeholders gather information from the same device, requesting various verifiable presentations. This can pose privacy challenges, but ZKPs can address this by concealing unnecessary data.

Moreover, when dealing with data custody involving IoT devices and the creation of VCs, it is crucial to consider security risks. Various types of side-channel attacks can be employed to obtain information about VCs. For instance, mechanisms on the Internet may track IoT devices and the data they generate, such as cookies, device fingerprints, or location information. The proposed scheme cannot prevent the use of these tracking technologies if they have been installed on the devices or the broker.

Another risk arises when links are embedded in VCs. While the credential itself is protected against tampering, its external content is not, making the links vulnerable to modifications by attackers. One way to mitigate this risk is by generating a hash of the external data and incorporating it into the credential. Another option to explore is using a private blockchain to store the data.

5 Experiments, Results, and Analysis

In this section, we present the results of the simulation tests and performance analysis conducted to assess the efficiency of our architecture.

5.1 Implementation

We utilized Java libraries for the implementation of our application. For the entities interacting with our application, we employed containers using Docker and Docker Compose [36] and Python scripts to generate telemetry data simulations. Below, we present the list of technologies employed.

1. Libraries to implement our application:
   - ACA-Py Java client [37] for instantiating cloud agent resources.
   - Indy SDK for creating DIDs and registering them on the blockchain.
   - Bouncy Castle [38] for secure connections with the broker using certificates.
   - Java Paho MQTT client [39] for subscribing to MQTT topics/messages.
2. Docker and Docker Compose for creating and running containers for entities:
   - The issuer agent, holder, and wallet in a PostgreSQL [40] database created using the ACA-Py library.
   - Von Network is like an Indy blockchain with four nodes and the ledger browser.
   - MongoDB [41] for storing data from IoT devices.
   - MQTT broker using Mosquitto [42].
3. Python scripts to simulate devices sending messages to the Mosquitto broker.

Through the developed application, it is possible to collect data via the broker while automatically issuing credentials. Additionally, we can store these collection-related pieces of information in a database, create DIDs that identify devices and collected data, and register them on the blockchain. Furthermore, the application enables connections with other agents and creates and lists schemas and credential models. We have implemented an API encompassing all these functionalities to streamline performance measurement tests. The application was developed using the Java programming language, and its source code is available on GitHub [43].

5.2 Results and Analysis

To conduct the tests, we utilized a single computer system equipped with an Intel Core i7-11700 processor, 16 GB of RAM, and running the Windows 11 Pro operating system. In this configuration, we measure the time required to register the DIDs of data collection on the Von network's blockchain. The registration of DIDs for devices follows a similar process, but it occurs only once for each device, unlike data registration, which can be resource-intensive. For this reason, we include this process in our tests. Subsequently, we record the time spent on credential issuance.

We conducted two independent tests, each involving the collection of 1000 samples. We calculated the means and standard deviations for each case. In the first test, we recorded DIDs on the blockchain for evidence identification, while the second test focused exclusively on credential issuance to another agent, excluding the registration of DIDs from the first scenario. Our objective was

to analyze the impact of blockchain registration on the architecture and how credentials behave without this additional process. Our evaluations encompassed measuring the time required for each process and monitoring RAM and CPU usage in both tests.

The results obtained are summarized in Table 2 below.

Table 2. Experimental Results

Test	Measure	Value
DID registration	Average Response Time	1503.059 ms
	Standard deviation	868.355 ms
	Average CPU usage	0,3%
	Average RAM usage	131 MB
VCs exchange	Average Response Time	18.891 ms
	Standard deviation	9.494 ms
	Average CPU usage	0.4 %
	Average RAM usage	171 MB

The CPU utilization percentages were measured using the Netbeans Profiler [44]. The request simulation was conducted using JMeter [45]. The first scenario took approximately 50 min and 7 s, while the second required 36 s. CPU and RAM usage monitoring was carried out during these periods. To establish a baseline, the application startup, which loads the libraries mentioned in Subsect. 5.1 and establishes connections with the agent, broker, and blockchain, took approximately 3 s, consumed 121 MB of RAM, and utilized 8% of CPU capacity.

Based on these results, we can highlight the following conclusions:

- Considerably longer times were observed in the DID registration processes, indicating a significant overhead on the ledger nodes when handling multiple requests.
- Regarding RAM and CPU usage, we observed that it is not resource-intensive. Therefore, even with an increase in the number of devices sending data, the application should be able to perform these operations without significant issues.
- Based on our experiments, we found that the maximum RAM required was approximately 180 MB. Furthermore, tests conducted with multiple credential exchanges in parallel do not significantly impact CPU and RAM usage.

In conclusion, the credential exchange process in the Gateway proved to be significantly faster when the DIDs registration in the data collection blockchain was not performed. This is because, in the credential issuance process, once the schema and credential definition are created and registered on the blockchain, this procedure does not need to be repeated. It can be executed only once to

generate numerous credentials, which will be signed with the public key stored in the wallet.

It is essential to note that our assessment does not address network latency since the tests are conducted on the same computer. Furthermore, outcomes may vary depending on the individual resources of each computer.

As a result, we conclude that the blockchain registration overhead will be the primary factor affecting performance. If evidence registration on the blockchain is necessary to compose the credential, this may pose a scalability challenge. Therefore, it is essential to evaluate the necessity of this step or consider alternative proposals aimed at reducing the overhead on the distributed ledger. However, the presented results are promising and suggest that issuing credentials on a large scale for IoT devices is feasible.

6 Conclusion

This study proposed an innovative Gateway architecture to integrate IoT devices with limited resources in Self-Sovereign Identity (SSI) technologies. The architecture enables data certification by facilitating the issuance of verifiable credentials (VCs) based on data collected from IoT environments and utilizing a distributed verifiable registry to identify devices. These attributes position it as a relevant solution with significant potential for applications requiring data ownership assurance in these environments. As a result, the proposed architecture ensures reliability, integrity, and traceability of information, crucial factors for data custody, especially in scenarios involving forensic areas and IoT devices.

Compared to the vast array of related works presented, our study stands out as a comprehensive solution for verifying data integrity in IoT devices, explicitly focusing on applying SSI to address these challenges. While numerous studies have tackled specific issues related to IoT, such as security, access control, and communication efficiency in various scenarios [19–28], our approach is unique in its exclusive concentration on the challenges related to resource-constrained IoT devices and data custody in these environments. This enables us to provide a reliable solution to enhance IoT applications and facilitate secure data sharing.

The robustness of the results and insights obtained from the performance metrics analysis validates the effectiveness and reliability of our architecture, as well as provides guidelines for future optimizations. In addition, future studies in this field will focus on implementing a client for our application running directly on IoT devices, exploring ways to optimize energy consumption and network utilization. Additionally, we consider the possibility of comparing our proposals with solutions in other domains and conducting scalability and application analyses in real-world scenarios.

References

1. Algarni, S., et al.: Blockchain-based secured access control in an IoT system. Appl. Sci. (Switzerland) **11**(4), 1–16 (2021). https://doi.org/10.3390/app11041772

2. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): a vision, architectural elements, and future directions. Futur. Gener. Comput. Syst. **29**(7), 1645–1660 (2013). https://doi.org/10.1016/j.future.2013.01.010
3. Arshad, H., Jantan, bin, A., Abiodun, O.I.: Digital forensics: review of issues in scientific validation of digital evidence. J. Inf. Process. Syst. **14**(2), 346–376 (2018). https://doi.org/10.3745/JIPS.03.0095
4. Shah, M., Saleem, S., Zulqarnain, R.: Protecting digital evidence integrity and preserving chain of custody. J. Digit. Forensics Secur. Law (2017). https://doi.org/10.15394/jdfsl.2017.1478
5. Sporny, M., Longley, D., Allen, C., Sabadello, M., Reed, D.: Decentralized identifiers (DIDs) v1.0. W3C, W3C Working Draft (2019). https://www.w3.org/TR/did-core/. Accessed 29 Sept 2023
6. Sporny, M., Noble, G., Burnett, D., Zundel, B., Longley, D.: Verifiable credentials data model 1.0. W3C, W3C Recommendation. https://www.w3.org/TR/vc-data-model. Accessed 29 Sept 2023
7. Hyperledger Indy. Hyperledger Foundation Projects INDY. https://www.hyperledger.org/projects/hyperledger-indy. Accessed 26 Sept 2023
8. Indy SDK. Hyperledger Foundation Projects INDY. https://github.com/hyperledger/indy-sdk. Accessed 20 Sept 2023
9. Hyperledger Aries. Hyperledger Aries Cloud Agent Python. https://github.com/hyperledger/aries-cloudagent-python. Accessed 29 July 2023
10. Verifiable Organizations Network (VON). https://github.com/bcgov/von-network. Accessed 02 Oct 2023
11. Allen, C.: The Path to Self-Sovereign Identity. [S.l.] (2016). http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html. Accessed 29 June 2023
12. Brunner, C., Gallersdörfer, U., Knirsch, F., Engel, D., Matthes, F.: DID and VC: Untangling Decentralized Identifiers and Verifiable Credentials for the Web of Trust (2021). https://doi.org/10.1145/3446983.3446992
13. Peer Did Method Specification. W3C. https://identity.foundation/peer-did-method-spec/index.html. Accessed 26 Sept 2023
14. Curran, S., Howard, C.: Becoming a Hyperledger Aries Developer. [S.l.] (2021). https://learning.edx.org/course/course-v1:LinuxFoundationX+LFS173x+3T2021/. Accessed 4 Dec 2022
15. Curran, S., Howard, C.: Introduction to Hyperledger Sovereign Identity Blockchain Solutions: Indy, Aries and Ursa. [S.l.] (2021). https://learning.edx.org/course/course-v1:LinuxFoundationX+LFS172x+2T2021. Accessed 1 Nov 2022
16. SOVRIN Foundation. Self-Sovereign Identity and IoT. [S.l.] (2020). https://sovrin.org/wp-content/uploads/SSI-and-IoT-whitepaper.pdf. Accessed 1 Oct 2022
17. Hyperledger Aries Explainer. Hyperledger Aries. https://github.com/hyperledger/aries. Accessed 29 Sept 2023
18. Hyperledger Ursa Explainer. Hyperledger Ursa. https://github.com/hyperledger/ursa. Accessed 23 July 2023
19. Fedrecheski, G., Rabaey, J.M., Costa, L.C.P., Calcina Ccori, P.C., Pereira, W.T., Zuffo, M.K.: Self-sovereign identity for iot environments: a perspective. In: Proceedings of the Global Internet of Things Summit, GIoTS 2020 (2020). https://doi.org/10.1109/GIOTS49054.2020.9119664
20. Luecking, M., Fries, C., Lamberti, R., Stork, W.: Decentralized identity and trust management framework for Internet of Things. In: IEEE International Conference on Blockchain and Cryptocurrency, ICBC 2020 (2020). https://doi.org/10.1109/ICBC48266.2020.9169411

21. Regueiro, C., Gutierrez-Agüero, I., Agüero, A., Anguita, S., de Diego, S., Lage, O.: Protocol for identity management in industrial IoT based on hyperledger Indy. Int. J. Comput. Digit. Syst. **12**(1), 2210142 (2022). https://doi.org/10.12785/ijcds/120153
22. Dixit, A., Smith-Creasey, M., Rajarajan, M.: A decentralized IIoT identity framework based on self-sovereign identity using blockchain. In: Proceedings of Conference on Local Computer Networks, LCN, pp. 335–338 (2022). https://doi.org/10.1109/LCN53696.2022.9843700
23. De Diego, S., Regueiro, C., Macia-Fernandez, G.: Enabling identity for the IoT-as-a-service business model. IEEE Access **9**, 159965–159975 (2021). https://doi.org/10.1109/ACCESS.2021.3131012
24. Kortesniemi, Y., Lagutin, D., Elo, T., Fotiou, N.: Improving the privacy of IoT with decentralised identifiers (DIDs). J. Comput. Netw. Commun. **2019** (2019). https://doi.org/10.1155/2019/8706760
25. Berzin, O., Ansay, R., Kempf, J., Sheikh, I., Hendel, D.: A troca de IoT. arXiv:2103.12131 (2021)
26. Terzi, S., Savvaidis, C., Votis, K., Tzovaras, D., Stamelos, I.: Securing emission data of smart vehicles with blockchain and self-sovereign identities. In: Proceedings of 2020 IEEE International Conference on Blockchain, Blockchain 2020, pp. 462–469 (2020). https://doi.org/10.1109/BLOCKCHAIN50366.2020.00067
27. Theodouli, A., Moschou, K., Votis, K., Tzovaras, D., Lauinger, J., Steinhorst, S.: Towards a blockchain-based identity and trust management framework for the IoV ecosystem. In: Proceedings of the Global Internet of Things Summit, GIoTS 2020 (2020). https://doi.org/10.1109/GIOTS49054.2020.9119623
28. Fotopoulos, F., Malamas, V., Dasaklis, T.K., Kotzanikolaou, P., Douligeris, C.: A blockchain-enabled architecture for IoMT device authentication. In: 2nd IEEE Eurasia Conference on IOT, Communication and Engineering 2020, ECICE 2020, pp. 89–92 (2020). https://doi.org/10.1109/ECICE50847.2020.9301913
29. Sovrin DID Method Specification. https://sovrin-foundation.github.io/sovrin/spec/did-method-spec-template.html. Accessed 27 Sept 2023
30. AnonCreds Specification. https://hyperledger.github.io/anoncreds-spec/. Accessed 29 Sept 2023
31. Official Documentation for the Indy SDK. Hyperledger Foundation Projects INDY. https://hyperledger-indy.readthedocs.io/projects/sdk/en/latest/docs/. Accessed 01 Oct 2023
32. Masood, F., Faridi, A.R.: Distributed ledger technology for closed environment. In: 2019 6th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, pp. 1151–1156 (2019)
33. Lux, Z.A., Beierle, F., Zickau, S., Göndör, S.: Full-text search for verifiable credential metadata on distributed ledgers. In: 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS), Granada, Spain, pp. 519–528 (2019). https://doi.org/10.1109/IOTSMS48152.2019.8939249
34. Light, R.A.: Mosquitto: server and client implementation of the MQTT protocol. J. Open Source Softw. **2**(13), 265 (2017). https://doi.org/10.21105/joss.00265
35. Aries RFC 0050: Wallets. https://github.com/hyperledger/aries-rfcs/blob/main/concepts/0050-wallets/README.md. Accessed 25 Sept 2023
36. Docker Community. https://www.docker.com/community/. Accessed 25 July 2023
37. ACA-Py Java Client Library. https://github.com/hyperledger-labs/acapy-java-client. Accessed 20 Sept 2023
38. The Bouncy Castle Crypto APIs. The Legion of the Bouncy Castle. https://www.bouncycastle.org/. Accessed 15 Sept 2023

39. Java Paho MQTT Client. Eclipse Paho Project. https://www.eclipse.org/paho/. Accessed 02 June 2023
40. PostgreSQL. PostgreSQL Global Development Group. https://www.postgresql.org/. Accessed 2 Ago 2023
41. MongoDB. https://www.mongodb.com/. Accessed 4 Ago 2023
42. Eclipse Mosquitto. https://mosquitto.org/. Accessed 2 June 2023
43. Self-Sovereign Identity Gateway for the Internet of Things. https://github.com/cristiandossantos/iot-ssi-gateway. Accessed 03 Oct 2023
44. Apache Software Foundation. Apache NetBeans. https://netbeans.org/. Accessed 02 July 2023
45. Apache Software Foundation. JMeter. https://jmeter.apache.org/. Accessed 06 July 2023