Tokenization of Ownership Management for Web-of-Things with Role-based Modeling

Orçun, Oruç, Uwe, Aßmann, Maliha Raja
TU Dresden, Dresden, Germany

Currently, the Internet of Things (IoT) is connected to the virtual world through the Web of Things (WoT), allowing efficient utilization of real-world objects with Internet technologies. The WoT facilitates abstract interaction between applications and connected IoT devices, allowing owners to switch between devices while using multiple ones. To achieve this, virtual assets in WoT devices can be tokenized through smart contracts and transferred using hashed proof as transactions within blockchain networks that support virtual currencies. The goal of Web of Things is to establish connectivity, interoperability, and integration among IoT devices using web standards and protocols, reducing reliance on device manufacturers. This enables easy integration of Web 3.0 cryptocurrency for device management. This study proposes a solution for WoT applications involving different cryptocurrency definitions. Finally, simulation results are presented to demonstrate the tokenization-based ownership transfer in the Web of Things.

1. Introduction

Large-scale networks in the Internet of Things (IoT) face challenges, such as fragmented monitoring and isolated data, which impede comprehensive observation. When adopting diverse IoT technologies for different purposes, fragmentation occurs due to varying architecture of each solution. Inventory monitoring involves managing and controlling stocks using various sensors distributed throughout a network. Scalability is essential for end-users to effectively utilize IoT solutions in their business operations. To address these issues, the Web of Things (WoT) has been introduced. It represents virtual objects as proxies for abstract entities linked to physical objects. Each „thing“ is defined with metadata, events, and properties, enabling communication and management of WoT devices through a messaging framework or design pattern.

Application development is challenging and complex when it comes to Internet of Things (IoT) devices due to the diverse standardization of programming interfaces and communication protocols among different IoT platforms. For example, Arduino, Raspberry Pi, and BeagleBone are three common development boards used in IoT programming, and developers must write applications specific to each board's specifications. Consequently, this poses difficulties when transitioning an application from one protocol (e.g., OPC-UA) to another protocol (e.g., COAP). One of the primary objectives of the Web of Things (WoT) architecture is to provide a unified framework that spans from micro controller-level devices to cloud-based applications [1].

Ensuring data and application ownership is vital in complex applications, such as supply chain simulations involving retailers, wholesalers, and manufacturers. In such simulations, product tags must be shared and authenticated among members. Radio Frequency Identification (RFID) technology is commonly used to tag products in warehouses or items on assembly lines, consisting of components like RFID tags, RFID readers, and RFID-tag database. Each asset is represented by an RFID tag registered in the database, which may contain ownership-related data. Applications utilizing RF tags facilitate the transfer of ownership between parties. Data ownership in the Web of Things is also another important factor to protect accessing, processing, or getting benefits from economic exploitation. For instance, parties of Web of Things should be in an agreement to initiate access, processing, or economic exploitation of data. Ownership can be thought of as control and data ownership including access, create, modify, package, sell or remove data, and access privileges to others [2].

Traceability and auditability are essential functions for ownership transfer within a single network. The list of involved parties should be registered in the network to ensure data ownership in the Web of Things (WoT) applications. Cryptocurrency items, operating directly on the blockchain layer through autonomous programming entities (smart contracts), can provide traceability and reliable ownership transfer for WoT. Tokens, representing the token economy using Web3.0 technology, serve as a conceptual representation of ownership. The distinctiveness of tokens economy, proof of ownership plays a significant role in determining the priority of the communi-

[1] https://www.w3.org/2015/05/wot-framework.pdf

[2] https://ori.hhs.gov/education/products/n_illinois_u/datamanagement/dotopic.html

cation environment between WoT nodes. Another notable aspect is asset tokenization, which converts tangible and intangible assets into traceable digital tokens. Each token type, whether representing a fraction or the entirety of ownership, enables manageable and trackable ownership of assets[3].

Main contribution and research questions: The main focus of this research is to introduce a transparent and tamper-proof dataset that provides traceable records for ownership transfer in Web of Things devices. In this specific application, the need for multiple data tags between parties is eliminated by utilizing non-fungible fractional tokens (Fractional NFTs) and fungible tokens. Additionally, the concept of Fractional NFTs can be integrated to role-based self-sovereign identity, abstracted by smart contracts, to fulfill particular roles such as Issuer, Verifier, and Holders.

This study demonstrates the integration of these elements to achieve a seamless and secure ownership transfer mechanism for Web of Things devices.

1.1 Research Questions

In this research study, we would like answer the following research question to struct the main objective of this thesis.

1) How can different cryptocurrency interfaces be integrated with Web of Things ownership entities?

2) How can self-sovereign identity features such as Issuer, Verifier, and Holder be implemented through smart contract programming in Web of Things solutions?

2. Background

2.1. Identity and Data Management

The establishment of secure and reliable interactions among WoT devices of utmost importance, necessitating the utilization of unique identities for devices and entities. Identity management solutions such as digital certificates, public-key infrastructure (PKI), or decentralized identity frameworks such as Self-Sovereign Identity (SSI) can be employed to fulfill this requirement. The present study primarily focuses on SSI, aiming to differentiate it from crpytocurrency-based interactions. It is imperative that Identity and Data Management adhere to principles of data ownership and governance, as authorized access could have detrimental effects on the entire WoT ecosystem. Therefore, the safeguarding of data privacy and content assumes paramount significance, necessitating the application of modern encryption techniques such as HTTPS or MQTT-TLS. While blockchain technology does offer a certain level of security, this aspect should not be disregarded.

2.2 Cross Domain Collaboration

In the realm of technological advancements, the Web of Things (WoT) emerges as a potent force, facilitating effortless interaction and collaboration among diverse entities across multiple domains. It empowers devices, platforms, and services from disparate domains to collaborate their efforts and work harmoniously towards the achievement of shared objectives. In order to foster cross-domain collaboration, it becomes crucial to establish a robust foundation and interoperability among data models, standard protocols and interfaces. In Figure 1 illustrates a fundamental interaction between ownership transfer and web of things ecosystem, exhibiting the interconnectedness and significance of these elements in the WoT landscape.

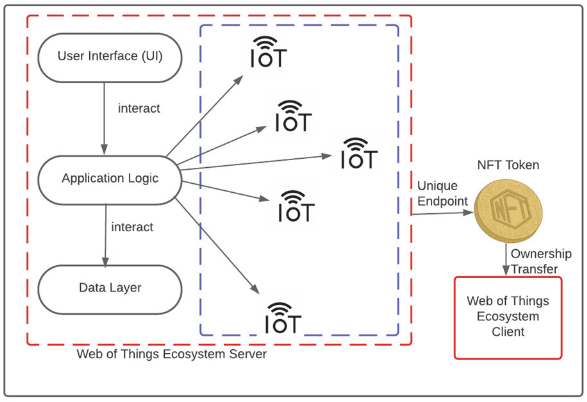

Fig. 1: Web of Things Ecosystem Demonstration Through Client and Server

2.3 Role-based Modeling

Role-based modeling elucidates how objects can take on diverse roles in multiple collaborations, effectively representing multiple identities. Due to the inherent nature of objects, each object must embody a single identity upon its creation. However, to represent distinct roles or multiple identities within a single object, we need to construct a player relationship through mixins, traits, design patterns (e.g. decorator, mediator, adapter, role object patterns) or subtyping. In the context of this study, roles are generated using modified mediator pattern with “lockNFT()” function, which facilitates the division of a Non-Fungible Token (NFT) into multiple holders through the involvement of an issuer and verifier. This approach enables the representation of various roles and identities within the context of the NFT ecosystem.

2.4 Democratizing Ownership

It refers to the concept of decentralizing ownership and empowering entities to have control other devices. These entities may have influence over the devices, data, and services within the WoT ecosystem. Individual nodes

[3] https://due.com/blockchain-asset-ownership/

should be empowered with greater control and autonomy over devices and services in the WoT.

2.5 Self-sovereign Identity

Self-sovereign identity minimizes the reliance on third parties and instead promotes a decentralized approach to private authentication storage, enabling individuals to manage their identities and access to them [1]. With this definition, it becomes apparent the smart contract technology can empower individuals with ownership and control over personal data related to ownership. Upon deploying a smart contract, true decentralization is achieved as individuals gain full control over the data layer of the smart contract. The use of smart contracts eliminates the centralization of control, as every node in the network must synchronize with the latest version of the smart contracts.

2.6 Types of Cryptocurrencies

Within the domain of digital currency, a range of token standards are available, including ERC-20, ERC-721, ERC-1155 and ERC-3475. These tokens can be utilized in single or multiple smart contracts, depending on their specific requirements and definitions. Through the facilitation of token transfers. Web of Things (WoT) devices can effectively monitor transactions between devices and allocate computing resources based on token expenditure. The tokenization of smart contracts also enables the development of decentralized applications (dApps). In the context of dApps designed for groups of IoT devices, the implementation of multiple replicated transaction ledgers becomes feasible without the need for a central authority.

The ERC-20 interface is widely employed across a range of scenarios within the blockchain industry. It provides a comprehensive set of functions that enable the efficient distribution of tokens within a blockchain network. These functions encompass obtaining the total token supply, verifying the balance of an account, managing allowances, executing token transfers, granting approval for token usage, and facilitating transfers between accounts. Consequently, it can be inferred that ERC-20 tokens should possess the capacity to retrieve the total token supply, evaluate the balance of designated accounts, facilitate seamless token transfers, and authorize token usage.

In response to the shortcomings of the ERC-20 token standard, the Ethereum community has proposed the ERC23 and ERC223 token standards. These proposed standards aim to address the following issues: lost tokens, lack of event handling, optimization of ERC20 address-to-contract communication, and disparities between Ethereum and Token Transfer mechanisms [4] [5].

Introducing an interface for managing various token types, including fungible, non-fungible, and semi-fungible tokens, ERC-1155 enables the deployment of a single contract that consolidates these token types. This consolidation eliminates the need for separate contracts associated with different token standards, such as ERC-20 for fungible assets and ERC-721 for non-fungible assets. The approach of consolidating token types within a single contract mitigates the issue of opcode bloat in the blockchain virtual machine. An illustrative example of leveraging this capability can be seen in the case of Gnosis [6], a company that utilizes conditional tokens to address multiple use cases while reducing gas costs for users by considering potential future outcomes in trading. Similarly, within the context of ownership transfer, this type of token holds potential for various scenarios involving the transfer of ownership within Web of Things (WoT) devices.

ERC-3475 is a standardized interface for contracts that handle multiple callable bonds. This standard entails a more intricate data structure than ERC-20, but it offers distinct functions to facilitate the reading and transfer of bond collections, as well as the issuance and redemption of bonds. By utilizing ERC-3475, it is possible to create numerous types of bonds within a single contract. Each bond is associated with a “classID”, which allows for the definition of new configurable token types.

ERC-725 and ERC-735 have been created to address blockchain-based identity solutions and involve the implementation of proxy smart contracts that can be managed by other smart contracts. These standards are specifically designed to cater to Self-Sovereign Identity use cases within blockchain applications. The key distinction between ERC-725 and ERC-735 is that the former represents the identity itself, while the latter represents the claims associated with the identity.

ERC-223 was introduced to address a significant bug in the ERC-20 standard for token exchanges. This bug was specifically related to the “*transfer()*” function within the blockchain network. In the case of transferring tokens to an externally owned account (EOA), the transfer could appear successful even if the EOA did not receive the tokens properly, potentially resulting in the token being permanently lost or burned. As a solution, ERC-223 proposed new standards for the “*transfer()*” function within the ERC-20 standard, aiming to rectify this issue.

[4] https://github.com/Dexaran/ERC223-token-standard

[5] https://github.com/iam-dev/ERC23

[6] https://www.gnosis.io/

ERC-667 seeks to combine the functionalities of both ERC-20 and ERC-223 standards, specifically focusing on enabling seamless token transfers and "*call()*" functions. Its primary objective is to facilitate token transfers through triggered contracts within a single transaction.

The ERC-721 standard is used to establish ownership of Non-Fungible Tokens (NFTs). Unlike ERC-20 tokens, NFTs cannot be treated interchangeably due to their unique properties. Essentially, ERC-721 serves as a token standard specifically designed for non-fungible assets. Common use cases for ERC-721 can include digital artwork, game collectibles, gaming characters, and art images.

ERC-173 establishes a standardized interface for contract ownership. Important aspects of ERC-173 include:

- The standard aims to minimize the number of functions in the interface to prevent contract bloat.
- ERC-173 provides backward compatibility.
- ERC-173 efficiently organizes the gas cost associated with smart contracts and this standard introduces a new approach for interacting with token contracts, ensuring compatibility with the ERC-20 Fungible Token Standard[7].

ERC-875 facilitates the use of non-fungible tokens by enabling the bundling of tokens into groups. This allows for peer-to-peer atomic transfer to occur within a single transaction. Essentially, atomic transactions guarantee that all internal transactions will either succeed or fail together.

ERC-918, known as the Mineable Token Standard, is a specification outlined in Ethereum Improvement Proposals. It relies exclusively on mining activities conducted through the Proof-of-Work concept. This standardization is commonly referred to as „Proof of Work Minting."

ERC-2615 is an extension to ERC-721 non-fungible token standard (NFT) to support rental and mortgage functions. This interface has been produced for real-world entities in the world such as mortgage agreements, real property with written agreements.

ERC-4626 is a standard for tokenized vaults that utilize ERC-20 tokens. It encompasses common functions like *transfer*, *transferFrom*, *balanceOf*, and *totalSupply*. This token type enables users to gain profits from their stakes. However, this token type does not address the security aspects between endpoints in a decentralized network. On the other hand, the Sealed NFT Metadata (ERC-3569) introduces a smart contract-based mechanism for immutable NFT metadata [8]. Both ERC-4626 and ERC-3569 are open standards, which means that they can be used in a mixed way through correct interfaces. This allows developers to create more complex and sophisticated applications that utilize the benefits of both standards.

All of these token types can be used in a mixed way through the appropriate interfaces. The specific implementation can vary based on requirements, but in this case, we have utilized the ERC-721 and ERC-20 standards to create the Fractional NFT.

3. Motivation and Challenges

In the context of supply-chain applications, ownership plays a crucial role in the transfer of goods between various participants such as retailers, manufacturers, and wholesalers. As an example, a tagged object can be moved from a manufacturer to a retailer. It is important to have visibility into the origin of the data source, the creation timestamp, and the expiration date at this stage. However, ensuring secure transfer of ownership between different data owners remains an ongoing research challenge. In this study, we aim to explore the feasibility of using different token types to address this challenge. By implementing a fully trustworthy ownership model using a block explorer, we can transparently track ownership through the Merkle tree data structure. Transactions between parties will be identifiable through the use of different token types. Last but not least, we would like to show the main case study to demonstrate technical challenges and theoretical limitations with regards to this research study.

4. Related Work

4.1 Web of Things and Ownership Transfer

Authors [2] describe for ownership transfer mechanism in the area of medical IoT devices [2]. According to authors, ownership principle can be transferred through immutable chained blocks by means of smart contracts addresses. Medical Internet of Things device owners can set some rules and conditions for access and modify the records pertaining to the Medical IoT device ownership [2]. Another study shows us how using blockchain technology can provide unique identifiers for IoT devices through records immutability [3]. In this study, while deploying an IoT system, the owner of the device dictates the transfer of ownership [3]. Transferring the device ownership will require a transfer process in a secure manner or device ownership can split between different owners. For instance, an IoT device can be held by tenants and a lender companies. According to the authors [3], large IoT infrastructures must be managed and controlled from a security perspective. One of the core problem in such big IoT systems is the lack of forward and backward secrecy with respect to the old and new owners. To solve this problem, the authors [3] offers a solution called BYODID (Bring-Your-Own-Device-Identity) to

[7] https://eips.ethereum.org/EIPS/eip-777

[8] https://eips.ethereum.org/EIPS/eip-3569

ensure a single user can have a transferable identity from one enterprise to another.

Each IoT device has a form of credentials that must be shared with a remote entity. According to the authors, ownership transfer is the process of updating the credentials on a protocol layer [6]. The ownership transfer process should be divided into three phases: deployment, ownership transfer preparation, and ownership transfer [6]. Even though this protocol design was designed for large IoT infrastructures, there is no real case study to evaluate the performance of ownership transfer protocol.

In order to protect privacy leaks, authors [8] proposed an automated ownership that would be triggered in the event of any ownership change. They proposed an automatic handling of ownership, which is the first system without user interaction during ownership change [8].

4.2. Security Challenges of Web of Things

Web of Things (WoT) is expected to make accessibility of smart things easy and promote by combining novel values according to the identity management such as ownership, identification, and social security [4]. Authors of the paper [4] concluded that authentication schemes like OAuth, JWT are not adequate to provide ownership transfer mechanism in WoTs.

Ownership transfer should be provided in a supply chain and changing ownership occurs when a wholesaler delivers tagged products to a retailer [5]. The authors of the paper [5] basically conducted a survey how to allow the secure and seamless transfer of ownership of RFID-tagged objects from one owner to another owner. As the authors stated, ownership transfer in IoT is generally supplied with Ownership Transfer Protocols (OTP), so one should take consideration of a particular protocol while deploying IoT applications regarding ownership transfer [5].

Burmester et al. [7] defines three steps of ownership transfer control, which are:

a) Preparation of a tag to be owned by another user.

b) Employing a trusted third party for a trustable link between current and new user.

c) Taking control of delivered tag when the protocol is completed.

In the conclusion of this paper, authors stated that preventing unauthorized tracking and secure ownership transfer are two major problems that need to be solved [7].

4.3. Token Types

Angelo et al. mention that security tokens are helpful to simulate the behavior of issuer, verifier, and holder [9]. Ownership transfer in cryptocurrency can be achieved through token contracts and *safe transfer* is a particular mechanism where token withdraw from an address or transferred to an address. While implementing this process, role-based authentication with lock control can be defined in the token contract [9]. Moreover, ownership transfer mechanism activities can be logged through cryptocurrency specific events [9].

Tokens can be categorized, according to di Angelo et al., into payment tokens, security tokens and utility tokens [9]. According to die Angelo et al., security token standards are proposed and discussed but not yet finalized [9]. Even in the Ethereum Mainnet, function signatures of security tokens are sparse, so one can assume that it is still an emerging technology yet [9].

5. Implementation

5.1. Design of the Use Case

The use case in Figure 2 has been successfully accomplished using Hardhat and the Solidity language package. These tools have been employed to showcase the fundamental functions of the use case, as well as the logic behind the issuer, verifier, and holder roles. In this implementation, the holder assumes the role of managing a specific data type within the Solidity language. As the holder, their responsibility is to safeguard the identification medium required by the verifier. On the other hand, the verifier is granted the authority to verify the identification medium on behalf of the issuer.

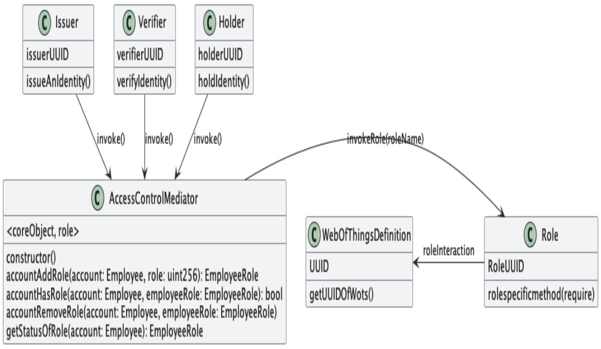

Fig. 2: Class Diagram for Access Control Mediator, Issuer, Verifier, Holder, Role, and WebOfThings Definition

In the use case scenario, inventory tracking in a warehouse has been implemented. According to the use case, the following activities are involved: receiving, inspection, putaway, storage, packaging and shipping. Basically, updating an inventory record and updating a storage record are accomplished by roles generated through the mediator pattern. Additionally, to achieve inventory tracking, fractional NFTs will be shared among WoT Devices.

6. Result

In order to assess the outcomes of proposed application in a qualitative manner, we aim to evaluate the incurred expenses related to both execution and deployment of the smart contract. By examining the costs associated with these aspects, we can obtain valuable insights into the financial implications of the application's implementation.

6.1. Quantitative Evaluation

Operation performance in terms of time calculation is crucial to understanding the efficiency of multiple contracts involved: the verifier, issuer, holder, and web of things identity processes. In Figure 3, deployment cost of self-sovereign identity is relatively big because smart contract role creation has a lot of interactions among each other. However, as can be seen in Figure 4, execution costs of smart contract are relatively high in the context design pattern role creation contract (Administrator contract).

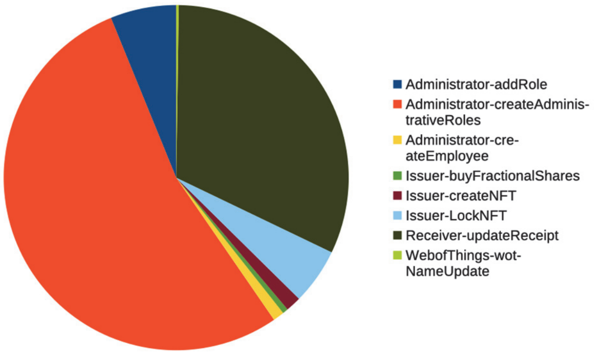

Fig. 3: Smart Contract Deployment Cost

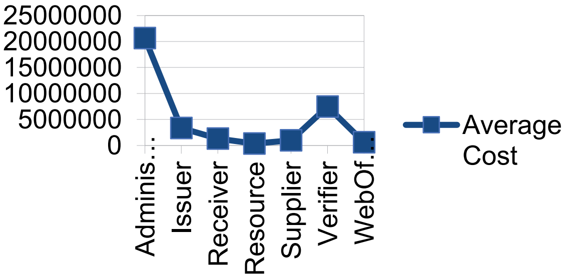

Fig. 4: Smart Contract Execution Cost

7. CONCLUSION

The metadata of the Web of Things (WoT) can be effectively represented through self-sovereign identity members, which can be verified by the verifier. This verification process ensures the smooth ownership transfer within the WoT ecosystem.

Additionally, the concept of fractional non-fungible tokens (NFTs) can be applied to Web of Things Tagged Resources, thereby establishing a robust self-sovereign identity system. Alternatively, a security token approach can also be explored; however, the current standard in this field is still in its nascent stages and not yet mature enough to be deployed for imitating self-sovereign identity use cases.

Roles within the system can be defined using various approaches, such as adapter, decorator, or role object patterns. However, it is important to note that the implementation of these patterns should be reliant on smart contract programming languages, as in the Solidity-Ethereum ecosystem have more constrained virtual machines and lack certain object-oriented properties. Moreover, most of the smart contract languages do not support object-orientation.

The code can be found under the GitHub link[9].

7.1. Future Work

Issuer and Verifier can be represented using various ERC interfaces or combination of them. The current implementation utilizes ERC-20 and ERC-721 interfaces to define the distribution token among the issuer, verifier, and holders. Moreover, the security token standard[10] can be integrated with one of the specific ERC interfaces. To minimize the cryptocurrency costs associated with the self-sovereign identity system, it is necessary to implement different design patterns and compare the final deployment results and function execution. This approach ensures optimal cryptocurrency efficiency throughout the system.

Acknowledgements / Information on sponsors

This work is funded by the German Research (DFG) within the Research Training Group Role-Based Software Infrastructures for continuous-context-sensitive Systems (GRK 1907)

Contact details

Orçun, Oruç: orcun.oruc@tu-dresden.de
Uwe, Aßmann: uwe.assmann@tu-dresden.de
Maliha, Raja: maliha.raja@tu-dresden.de

[9] https://github.com/zointblackbriar/Paper-Code/tree/main/Tokenization-Of-Ownership-Management-for-Web-of-Things-new

[10] https://github.com/SecurityTokenStandard/EIP-Spec

Literature references

[1] Alblooshi, M.; Salah, K. and Alhammadi, Y. (2018). Blockchain-based ownership management for medical IoT (MIoT) devices,: 151-156.

[2] Gunnarsson, M.; Gehrmann, C.; Furnell, S.; Mori, P.; Weippl, E. and Camp, O. (2020). Secure Ownership Transfer for the Internet of Things., : 33-44.

[3] Sardar, R. and Anees, T. (2021). Web of things: security challenges and mechanisms, IEEE Access 9 : 31695-31711.

[4] Samaila, M.; Neto, M.; Fernandes, D.; Freire, M. and Inácio, P. (2018). Challenges of Securing Internet of Things Devices: A survey, Security and Privacy 1.

[5] Omar, A. S. and Basir, O. (2018). Identity management in IoT networks using blockchain and smart contracts, : 994-1000.

[6] Burmester, M.; Munilla, J.; Ortiz, A. and Caballero-Gil, P. (2017). An RFID-Based Smart Structure for the Supply Chain: Resilient Scanning Proofs and Ownership Transfer with Positive Secrecy Capacity Channels, Sensors 17.

[7] Khan, M. S. N.; Marchal, S.; Buchegger, S. and Asokan, N. (2019). chownIoT: enhancing IoT privacy by automated handling of ownership change, Privacy and Identity Management. Fairness, Accountability, and Transparency in the Age of Big Data: 13th IFIP WG 9.2, 9.6/11.7, 11.6/SIG 9.2. 2 International Summer School, Vienna, Austria, August 20-24, 2018, Revised Selected Papers 13 : 205-221.

[8] Di Angelo, M. and Salzer, G. (2020). Tokens, types, and standards: identification and utilization in Ethereum, : 1-10.

[9] Di Angelo, M. and Salzer, G. (2021). Towards the identification of security tokens on Ethereum, : 1-5.