Annals of Telecommunications
https://doi.org/10.1007/s12243-024-01032-6

RESEARCH

Digital credentials management system using rejectable soulbound tokens

Rosa Pericàs-Gornals[1] · Macià Mut-Puigserver[1] · M. Magdalena Payeras-Capellá[1] · Miquel Á. Cabot-Nadal[1] · Jaume Ramis-Bibiloni[1]

Received: 11 January 2024 / Accepted: 25 March 2024

Abstract
Digital credentials are being issued by authorized entities to facilitate the digital identification of their users. Blockchain offers some inherent features that are highly advantageous for the management of credentials. Non-fungible tokens, or NFTs, might seem to be a perfect fit for the implementation of digital credentials. However, some crucial requirements for credentials are the non-transferability of the credential and that the authorized entity should receive explicit acceptance from the user who will own the new credential, which are features lacking in the current NFTs. This paper introduces a management system focused on issuing digital access credentials, enhancing traditional features by enabling the association of terms and conditions (T&C) during issuance and providing users with non-repudiation of reception evidence upon acceptance. Leveraging an enhanced version of the soulbound tokens (SBTs), called RejSBTs, introduced in our previous work, the new system guarantees non-repudiation of reception and origin proofs. Furthermore, we provide a detailed implementation of the system, including solidity smart contracts, accompanied by a comprehensive cost and security analysis.

Keywords Digital credentials · Identity · Blockchain · Soulbound tokens

1 Introduction

Since many years ago, the digitalization of assets has become increasingly prevalent, and the concept of digital identity has also kept up with this trend. Today, a multitude of digital credentials are being issued by entities to facilitate the digital identification of their users. In particular, a digital credential can be defined as "the digital equivalent of paper documents, plastic tokens and other tangible objects issued by trusted parties" [1].

✉ Macià Mut-Puigserver
macia.mut@uib.cat

Rosa Pericàs-Gornals
rosa.pericas@uib.cat

M. Magdalena Payeras-Capellá
mpayeras@uib.cat

Miquel Á. Cabot-Nadal
miquel.cabot@uib.cat

Jaume Ramis-Bibiloni
jaume.ramis@uib.cat

[1] Universitat de les Illes Balears, Palma, Spain

Digital credentials can be classified into three main categories:

- **Identity credentials** represent an individual's identity, such as digital passports or driving licenses.
- **Academic and professional credentials** validate the academic and/or professional achievements of the subject, such as a bachelor's or master's degree.
- **Access credentials** provide authorization for access in digital systems or platforms, such as a username and password.

The adoption of blockchain technology is an increasingly used solution to implement secure applications with digitized assets. Blockchain offers some inherent features that are highly advantageous for such applications. These features are data immutability, transparency, security achieved through cryptographic techniques, and traceability. With these inherent features, blockchain technology provides an important base for ensuring the integrity and security of digitized assets.

Ethereum or compatible blockchain networks have already defined relevant standards that greatly fit in the implementation of these types of applications. An example is the ERC-721 standard, which defines non-fungible tokens (NFTs). Unlike the ERC-20, which defines fungible tokens, NFTs are unique and cannot be interchanged. Specifically, NFTs serve as digital representations of real-world assets, representing the value of goods or services, or offering utility based on their intrinsic worth.

At first glance, NFTs might seem to be a perfect fit for the implementation of digital credentials, due to their ability to digitally represent real-world assets. Moreover, there already exists a multitude of implementations of digital credentials based on the usage of NFTs. However, in the case of digital access credentials, a crucial requirement is the non-transferability of the credential and the explicit acceptance from the user who will own the new credential. These credentials need to restrict transferability after issuance to prevent unauthorized access by individuals not originally intended and also need to allow the intended holder to decide whether to accept or decline the reception of the digital credential.

To address the first requirement, Buterin et al. [2] recently introduced a variant of NFTs known as soulbound tokens (SBTs). The primary objective of SBTs is to ensure the non-transferability of assets. SBTs are specifically designed to be bound to a user's wallet, which the authors refer to as "Souls". Consequently, this innovative type of token aligns more closely with the fundamental requirement of digital access credentials, by imposing the necessary limitations on transferability.

However, SBTs currently lack a feature that enables users to accept or reject the reception of the tokens and, moreover, there is still no standard implementation of Buterin et al. proposal. In our previous work [3], we introduced the Rejectable NFTs (RejNFTs), an improvement of the ERC-721 standard, where we implemented the ability to selectively reject this kind of token.

This journal paper is an invited extended version of a conference paper [4] presented at the 7th Cyber Security in Networking Conference, which took place in Montreal, Canada, on October 16–18, 2023. The conference paper formed the basis for further research and expansion, resulting in the content presented herein. This paper presents a management system based on a new protocol that is focused on the issuance of digital access credentials, with the additional ability to accept the associated terms and conditions (T&C) that the intended holders must accept when receiving the credentials. We propose and use RejSBTs, a new kind of token, to represent the credentials and associated T&C, providing non-repudiation of reception and origin evidence. The extended version of the paper includes an enhanced description of the protocol together with new work performed in order to evaluate the proposal. After the implementation of the protocol it has been evaluated both to prove that the protocol achieves the desired properties for this kind of service and to prove that the resulting implementation is viable in terms of performance.

The rest of the paper is organized as follows. Section 2 introduces the essential properties of digital access credentials and the state of the art of the subject. Section 3 outlines the main contribution of our system. Subsequently, in Section 4, the protocol for the issuance of the digital access credentials is presented. In Section 5, we provide the smart contracts implementation, and Section 6 provides a Security Analysis of the defined and implemented protocol. Section 7 evaluates the execution costs associated with the implementation. Finally, in Section 8, the conclusions of the designed protocol and future work are presented.

2 Properties and state of the art

Digital credentials were first proposed by Brands in 1993 [5] as a secure means to represent real-world objects in a digital format. The National Institute of Standards and Technology (NIST) defines digital identity as "The unique representation of a subject engaged in an online transaction. A digital identity is always unique in the context of a digital service but does not necessarily need to uniquely identify the subject in all contexts. In other words, accessing a digital service may not mean that the subject's real-life identity is known" [6].

Traditionally, digital access identities or credentials have been centrally managed, limiting the control that identity holders had over them [7]. However, with the introduction of blockchain technology and other decentralized frameworks, digital access credentials [8, 9] and identities [10] have gained the self-sovereign identity (SSI) feature [11]. SSI empowers credential holders with complete control over their data, enabling the credential to possess value on its own, without the intervention of the issuer.

Currently, there are several proposed protocols for the management of digital credentials. For instance, Herbke et al. [12] presents a protocol that applies the self-sovereign identity paradigm to student digital credentials.

A similar type of protocol for digital credentials management is the one focused on digital certificates management. In their work, Reza et al. [13] propose a blockchain-based framework for managing traditional paper-based certificates or credentials in a distributed ledger. The proposed framework incorporates a secure storage system, granting exclusive access to authorized parties, and thereby obviating the necessity for third-party involvement in certificate verification. Meanwhile, Eltuhami et al. [14] propose a novel approach to certificate management systems that utilizes non-transferable NFTs. However, a limitation of their proposal lies in the absence of a mechanism enabling the receiver user to reject the transfer of the token, potentially leading to scenarios where malicious users associate unwanted credentials with individuals, compromising their identity.

In [15] Hunhevicz et al. the authors investigate the usage of decentralized access methods using Web3, differentiating between two approaches: role-based and token-based. The first approach is managed by the user address, and the second one is managed by the issuance of a token. As the authors mention, the token-based approach makes access more flexible as the user can easily change the address holder of the token and the access rights. However, the ability to easily send access rights between addresses introduces a security gap where the holder could send the token to an incorrect address and thus lose access rights and data privacy.

In contrast, numerous ongoing projects are currently addressing the management of digital credentials. BCDiploma, for instance, introduces a new format of digital credentials to the traditional ones emitted over the blockchain, known as Web3 digital credentials [16]. BCDiploma offers three distinct formats of Web3 digital credentials: NFTs, SBTs, and verifiable credentials. Notably, none of these formats explicitly specifies the option for selective reception, a critical feature to prevent the reception of unwanted certificates.

Another noteworthy project is the digital credential infrastructure being developed by the digital credentials consortium. As outlined in their white paper [17], this consortium is actively engaged in developing an infrastructure for digital credentials related to academic achievements. Analyzing the credential issuance process, both the issuer and the learner (referred to as the digital credential holder) must undertake multiple tasks to facilitate the selective reception of the newly issued credential. The learner is responsible for two tasks, and the issuer must execute four tasks, potentially contributing to an extended issuance timeline for the new credential.

Considering the current state of the art and the characteristics of digital credentials, we have generated a list of desired properties for digital credentials. This list has been created processing the proposed lists of properties extracted from relevant papers [18–21].
The desired features of a digital access credentials system are:

- **Integrity and immutability**. The data within the issued digital credential must remain intact and unaltered, ensuring that the data cannot be modified or tampered without detection. Furthermore, the data should not be deletable, and a historical record of it should be maintained.
- **Availability**. The digital credential and the associated management system should always be accessible and usable.
- **Transferability of evidence**. The evidence associated with the issued digital credential should be capable of being presented or transmitted to relevant parties, during authentication or authorization processes.
- **Interoperability**. The digital credential format should seamlessly work with other systems or platforms.
- **Non-repudiation**. The entity issuing the digital credential should not be able to deny their origin and the digital credential holder should not be able to deny the reception.
- **Self-sovereign identity**. Should enable the holder to manage and control the digital credentials, without relying on centralized authorities.
- **Selective reception**. Should allow the intended holder to decide whether to accept or decline the reception of the digital credential.

3 Contribution

In general, the current main proposals are defined to manage digital credentials represented as certificates or digitize traditional paper-based credentials. Our paper proposes a blockchain protocol to manage digital access credentials and associated T&C, providing the holder access to systems and platforms. Additionally, we focus on two important aspects that help to improve previous proposals. On one hand, NFTs and SBTs, as they are defined in their standards, lack the functionality to allow the intended holder to reject the transfer (and, as a consequence, the reception) of the token. Thus, if NFTs or SBTs are used to deliver digital credentials, then the intended holder cannot reject the reception. This way, all kinds of digital credentials could be associated with the user's identity, even without their consent. With our proposal of a new kind of token, RejSBT, the credentials have to be explicitly accepted by the intended holder (providing non-repudiation of reception evidence) and, in addition, they cannot be transferred to third users.

On the other hand, in many applications, the delivery of digital credentials is associated with the acceptance of T&C of the issued credential. For this reason, we propose a protocol that includes the acceptance of the T&C when the credential is issued. The proposed protocol has been implemented and evaluated in terms of security and performance.

4 Protocol description

In this paper, we present a new protocol to send credentials and T&C to users in the form of RejSBTs. This section presents an overview of the proposed protocol, describing the participating parties and their interactions.

The proposal considers the following actors and roles:

- **Issuer** (I): A user or entity responsible for issuing digital credentials and establishing the T&C. This authorized

entity identifies the intended holder of the token, action out of scope of the current protocol. It is considered that the issuer using any on-chain or off-chain mechanisms, possesses all necessary information from the holder to facilitate the correct identification within the system.
- **Holder** (H): A user that receives the transfer of the digital credentials and T&C. Her main action is to decide whether to accept or reject the reception of the digital credential and the associated T&C.
- **Verifier** (V): user, entity, or system that needs to validate the digital credential provided by H. It just needs to access the digital credential content and check the stored data.

According to the above user descriptions, the issuer is responsible for the identification of the holder and it has to associate each holder to a blockchain address. For instance, if the issuer is a University, then this institution is in charge of associating each student with a blockchain address. This paper is not focused on this feature; however, the issuer has to follow the standard guidelines in technology that are described in documents such as [22] where it is provided a set of requirements for enrollment and identity proofing of applicants in order to avoid or mitigate impersonation, and either compromise or malfeasance of the infrastructure provider.

The digital credential is composed of several key components. Firstly, it includes the credentials and T&C data, which encapsulate the necessary information related to the digital credential being shared. Additionally, the digital credential incorporates a $deadline$, which defines the duration within which H has the opportunity to accept the reception of the token. This $deadline$ ensures that there is a defined period for the holder to review and decide on the acceptance of the digital credential. Finally, it can optionally include an $expiry$ date, acting as a boundary, indicating the point at which the accepted digital credential will expire. Once the $expiry$ date is reached, the token becomes invalid, emphasizing the temporal aspect of the RejSBT and ensuring that the digital credential within it has a limited lifespan.

The inclusion of an expiry date in the digital credentials system is an optional feature, as there are two distinct types of digital access credentials:

- **Permanent credentials**: characterized by their infinite durability, providing the holder with unrestricted access indefinitely. For instance, a professor on an academic platform may possess permanent credentials, ensuring perpetual access to their educational materials.
- **Temporal credentials**: characterized by their finite duration. They come with an explicit $expiry$ date, defining a finite period during which the holder retains access to the system. For example, a learner in an academic platform loses access to the online materials upon the completion of the course period, when the $expiry$ date is reached.

The presented protocol for sending digital credentials is composed of the following group of steps, which are depicted in Fig. 1:

1. **Issuance of the digital credential**: I creates the content of the credentials and defines the T&C to be associated with their acceptance. If deemed necessary or important, I uploads both the credentials and the T&C to a decentralized storage system, such as IPFS, although the specific details of this upload process are beyond the scope of the protocol described in this paper. To fulfill international requirements, according to the W3C recommendations for verifiable credentials [23] the language and also the base direction of the text of the terms must be taken into account. For this reason, different versions of the T&C in different languages can be stored in IPFS.
   Next, I mints a new token which is bonded to the credentials and T&C. During this minting process, I also defines two crucial parameters: the $deadline$ and the optional $expiry$ date.
2. **Holder acceptance**: if H wants to accept the received digital credential, she executes the acceptance function, within the defined $deadline$. By executing the acceptance function, H formally indicates the consent and agreement to the received credentials and associated T&C.
3. **Holder rejection**: if H does not want to accept the received credentials, she can actively reject the transfer

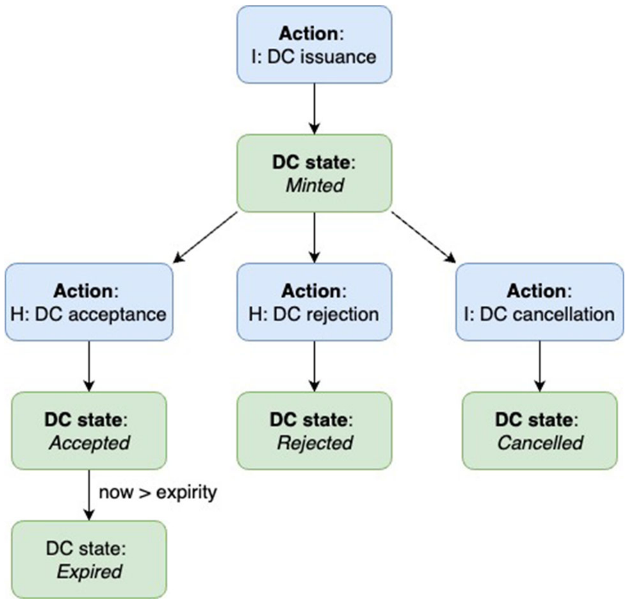


Fig. 1 States diagram of the protocol

or can simply do nothing, since the credentials will automatically become invalid when the defined *deadline* is reached.

4. **Issuer cancellation**: if *H* has neither accepted nor rejected the credentials, *I* retains the option to cancel the transfer. In this situation, *I* has the authority to cancel the token and nullify the transfer of the digital credential.
5. **Verifier's validation**: once *H* has accepted the reception of the digital access credentials just checking with *H*'s address, *V* can verify the ownership of the digital credential, all the metadata linked on it, and therefore can give the suitable right to *H* according to the credential verification result. Of course, if the verification is carried out after the *expiry* date, the system will return *expired* as the value of the digital credential status.

The core protocol, as currently depicted, lacks encryption measures, as it focuses on digital access credentials, which mainly do not require any personally identifiable information or any other sensitive data. But, it's imperative the alignment with GDPR regulations for any integration involving the presented protocol.

In this protocol, we define the essential steps for a digital access credentials system. However, it would be interesting to consider extending the protocol to address scenarios where credential revocation may be required due to failures in the verification processes or when a dishonest behavior by the holder is detected, contrary to the accepted terms and conditions. Moreover, it can be considered the addition of a provision that allows the possibility for the holder to cancel the digital credential after acceptance (in case he changes his mind). These aspects will be explored further as part of future research, as discussed in Section 8.

5 Implementation

In order to implement, test, and evaluate the performance of the protocol described in the previous section, we developed a set of smart contracts using the solidity programming language. These smart contracts, along with their corresponding tests, are available in a dedicated GitHub repository.[1]

For the implementation of the digital access credentials protocol, we will use a RejSBT smart contract to store all the digital access credential data. Following the proposal made in [3], both the issuer and the holder will possess the ability to control the transfer of the digital credential. Specifically, the issuer will be able to cancel the transfer of the digital credential, while the holder will be able to decide whether to accept or reject the token transfer. These functionalities will be subjected to a predetermined deadline set by the digital credential issuer.

```
// Mapping from token ID to transferable owner
mapping(uint256 => address) private _transferableOwners;
```

Listing 1 mapping of transferable owners

As we have previously mentioned, due to the lack of standard implementation of SBT, to implement our RejSBT we get the code of an NFT (ERC-721), and we modify it just maintaining the functions that do not involve or are related to the transfer functionality, except the `Mint` and `Burn`, as these functions are essential for the creation and destruction of the SBTs. According to the protocol proposed in [3], for the RejSBT we also integrate the rejectable functionality.

In order to enable the rejection by *H* of a new SBT, it is necessary to modify the existing `mint()` function. This removes the direct execution of the SBT transfer and defers it until *H* accepts it. For this reason, we introduce a new mapping called `_transferableOwners`, as presented in Listing 1. This mapping is responsible for storing the owner to whom *I* wants to transfer the new token. By utilizing this mapping, the minting process will only set the desired ownership transfer without directly transferring the token. The ownership transfer will occur once *H* accepts the transfer request.

Listings 2 and 3 show the implementation of the digital credential issuance functionality. This implementation consists of a public function (Listing 2) and a private function (Listing 3). The private function is inherited from a smart contract that defines the generic RejSBT with a deadline, which restricts the time frame during which acceptance, rejection, or cancellation of the SBT is permitted.

In order to execute the digital credential issuance functionality, *I* executes the *mint* public function (Listing 2). This function performs several checks to ensure the proper definition of the digital credential. Specifically, it verifies that at least the *credentials* or *terms* are provided, and validates the *expiry* date and *deadline* values.

In addition, if the digital credential includes an expiry date, its value must be at a later date than the defined deadline time frame. This condition is crucial to prevent the acceptance of expired digital credentials. Contrarily, if the digital credential does not need an expiry date, the *expiry* value must be set to 0. By applying these validations, the issuance process ensures that the digital credential is correctly defined, and maintains the integrity and validity of the protocol.

Within the public *mint* function, the private *_mint* function is called, and finally, the digital credential metadata is populated. The metadata includes information on the credentials, T&C, and the given expiry date.

[1] https://github.com/secomuib/credentials-soulboundtoken-main

```
function mint(
    address to,
    uint256 deadline,
    string memory credentials_,
    string memory terms_,
    uint256 expiry_
) public returns (uint256) {
    require(
        (keccak256(abi.encodePacked((credentials_))) !=
            keccak256(abi.encodePacked((""))) ||
            keccak256(abi.encodePacked((terms_))) !=
            keccak256(abi.encodePacked(("")))),
        "CredentialsRejectableSBT: credentials and terms
            are empty"
    );

    require(
        expiry_ != 0 ? deadline < expiry_ : deadline >
            expiry,
        "CredentialsRejectableSBT: incorrect expiry
            date value"
    );

    uint256 tokenId = _tokenIdCounter.current();
    _tokenIdCounter.increment();
    _mint(to, tokenId, deadline);

    credentialsData[tokenId] = CredentialsData({
        credentials: credentials_,
        terms: terms_,
        expiry: expiry_
    });

    return tokenId;
}
```

Listing 2 Public function for the digital credential issuance

The private *_mint* function is responsible for performing the following three checks:

- The holder address (*to*), representing *H* address, must not be the Zero address. This ensures that a valid *H* address is provided for the digital credential issuance.
- The given *tokenId* must not have been already minted, guaranteeing that the *tokenId* is not reused for multiple digital credentials, ensuring uniqueness and preventing conflicts.
- The value assigned to the *deadline* must be later than the current time. This ensures that the *deadline* is in the future, allowing sufficient time for acceptance, rejection, or cancellation of the digital credential transfer.

Once these verifications have been successfully executed, the used mapping is populated:

- `_minters` contains the address of the digital credential issuer (I).

```
function _mint(
    address to,
    uint256 tokenId,
    uint256 deadline
) internal virtual {
    require(
        to != address(0),
        "RejectableSBTDeadline: mint to the zero address"
    );
    require(
        !_exists(tokenId),
        "RejectableSBTDeadline: token already minted"
    );
    require(
        deadline > block.timestamp,
        "RejectableSBTDeadline: deadline expired"
    );

    // _msgSender() is the issuer
    _minters[tokenId] = _msgSender();
    _transferableOwners[tokenId] = to;
    _deadlines[tokenId] = deadline;
    _states[tokenId] = State.Minted;

    emit TransferRequest(_msgSender(), to, tokenId);
}
```

Listing 3 Private function for the digital credential issuance

- `_transferableOwners` stores the address of the digital credential holder (H).
- `_deadlines` contains the given deadline of the digital credential.
- `_states` stores the state value of the digital credential, which is set to *minted* in this case.

It is important to note that the execution of the `mint()` function does not trigger the RejSBT transfer; it will take place in the `acceptTransfer()` function, as shown in Listing 4.

For the acceptance of the digital credential transfer, we have introduced a new function called `acceptTransfer()` (Listing 4), and for the rejection, we have introduced a function called `rejectTransfer()` (Listing 5). Both functions share the same verifications:

- H address must be included in the `_transferable-Owners` mapping, ensuring that H has been proposed as the transferable owner of the given *tokenId*.
- The current execution time must be before the defined *deadline*, ensuring that the acceptance or rejection of the transfer is executed within the allowed time frame.
- The given *tokenId* must not have been already minted, ensuring that the acceptance or rejection applies to a valid digital credential.

```
function acceptTransfer(uint256 tokenId) public virtual
     override {
        require(
            _transferableOwners[tokenId] == _msgSender(),
            "RejectableSBTDeadline: accept transfer
                caller is not the receiver of the token"
        );
        require(
            _deadlines[tokenId] > block.timestamp,
            "RejectableSBTDeadline:deadline expired"
        );
        require(
            _states[tokenId] == State.Minted,
            "RejectableSBTDeadline: token is not in
                minted state"
        );

        address from = minterOf(tokenId);
        address to = _msgSender();

        _balances[to] += 1;
        _owners[tokenId] = to;
        _states[tokenId] = State.Accepted;
        // remove the transferable owner from the mapping
        _transferableOwners[tokenId] = address(0);

        emit AcceptTransfer(from, to, tokenId);
    }
```

Listing 4 Function to accept the transfer of the digital credential

```
function rejectTransfer(uint256 tokenId) public virtual
     override {
    require(
        _transferableOwners[tokenId] == _msgSender(),
        "RejectableSBTDeadline: reject transfer caller is
             not the receiver of the token"
    );
    require(
        _deadlines[tokenId] > block.timestamp,
        "RejectableSBTDeadline: deadline expired"
    );
    require(
        _states[tokenId] == State.Minted,
        "RejectableSBTDeadline: token is not in minted
            state"
    );

    address from = minterOf(tokenId);
    address to = _msgSender();

    _states[tokenId] = State.Rejected;
    _transferableOwners[tokenId] = address(0);

    emit RejectTransfer(from, to, tokenId);
}
```

Listing 5 Function to reject the transfer of the digital credential

```
function cancelTransfer(uint256 tokenId) public virtual
     override {
    require(
        minterOf(tokenId) == _msgSender(),
        "RejectableSBTDeadline: cancel transfer caller is
             not the minter of the token"
    );
    require(
        _deadlines[tokenId] > block.timestamp,
        "RejectableSBTDeadline: deadline expired"
    );
    require(
        _states[tokenId] == State.Minted,
        "RejectableSBTDeadline: token is not in minted
            state"
    );

    address from = minterOf(tokenId);
    address to = _transferableOwners[tokenId];

    require(
        to != address(0),
        "RejectableSBTDeadline: token is not transferable
            "
    );

    _states[tokenId] = State.Cancelled;
    _transferableOwners[tokenId] = address(0);

x    emit CancelTransfer(from, to, tokenId);
}
```

Listing 6 Function to cancel the transfer of the digital credential

By implementing these verifications, both `acceptTransfer()` and `rejectTransfer()` functions to ensure that the appropriate conditions are met for the acceptance or rejection of the digital credential.

Once the compliance of these conditions is verified, the `acceptTransfer()` function proceeds with the transfer of $tokenId$ to the new owner by changing the ownership to the new address and removing `_transferableOwners` data of $tokenId$. On the other hand, the `rejectTransfer()` function only removes the `_transferableOwners` data for $tokenId$. In this case, there is no ownership transfer, as the transfer has been rejected.

Finally, Listing 6 provides the implementation to allow I to cancel an opened proposal, if H has not yet called neither the `acceptTransfer()` nor the `rejectTransfer()` functions. In such cases, I can execute the `cancelTransfer()` function. This function includes the following verifications:

- The caller of the function must be H, the user who started the issuance of the digital credential, ensuring that I is the user who executes the cancellation of the digital credential issuance.
- The current execution time must be before the defined *deadline*, ensuring that the cancellation is executed within the allowed time frame.

- The given $tokenId$ must not have been already minted, ensuring that the cancellation applies to an already non-minted digital credential.
- The given $tokenId$ must have some proposed transferable owner, ensuring that there is an already opened proposal for the $tokenId$ issuance.

In the case, where the verifications have been executed successfully, the `cancelTransfer()` function removes the `_transferableOwners` data for the given $tokenId$.

The implementation presented in this section achieves the main functionalities defined in the protocol.

6 Security analysis

In this section, the fulfillment of the desired features of a digital access credentials system, described in Section 2, will be analyzed:

- **Effectiveness.** The system for digital credentials management presented in this paper is effective; thus, all parties will receive the expected items if they behave according to the protocol.

  On one hand, issuers have the interest in certifying certain achievements of their users or they simply want to attest to some participation or membership in an organization or activity. On the other hand, holders aim to earn the certification according to their actions, knowledge, or affiliations. Moreover, they require the capability to reject or accept any certification in order to avoid undesirable bonds. The effectiveness of this protocol addresses these features because, in order to create a new *Digital Credential*, the issuer generates a new token and executes the functions following the specifications of the protocol. If the holder accepts the transfer of the token, the digital credential will be generated, and the ownership of the token will serve as proof of the delivery of the digital credential, along with the acceptance of the T&C. Upon completion, if H has followed the *Holder acceptance* protocol, H will have the token representing the credentials. In Fig. 1, the *accepted* state represents this situation.
- **Integrity and immutability**. The digital credential and its associated T&C must remain intact and unaltered, ensuring that the data cannot be modified or tampered without detection, until their expiration date, if applicable.
  To achieve reliability in a Digital Credential system, the protocol must ensure that the credentials remain unchanged since their creation. In the protocol, data is stored in the blockchain in the form of a Rejectable SoulBound Token (RejSBT). Consequently, the digital credential cannot be modified, ensuring its integrity. Moreover, all transactions related to the creation and acceptance of the digital credential and its associated T&C are stored in the blockchain. Due to the inherent features of the blockchain, the data stored in it is immutable.
- **Temporal parameters: timeliness and timestamping.** A successful digital credential delivery will always be completed before the deadline d.
  In security protocols, it is important to ensure that protocol actors can complete their execution within a finite time, and the system has to guarantee the fulfillment of time constraints among different protocol events. Regarding the protocol presented in this paper, if the delivery is not successful, we have different situations depending on how the exchange has been performed. If H does not accept the notification and executes the *Holder rejection*, then the delivery will immediately lead to the *Rejected* state. If H neither accepts nor takes any action, the delivery will remain in *Minted* state unless the issuer I executes a *Cancellation* before the deadline d. After the acceptance of the digital credential, its validity will extend until the expiration date or indefinitely, depending on the value of *expiry*. The Smart Contract managing the system on the blockchain will always verify the deadline before returning the digital credential's state in order to enforce the timeline parameters. An expired credential is represented by the state *Expired*. This way, before the deadline, the state of the delivery (either *Minted*, *Accepted*, *Rejected*, or *Cancelled*) will be known and cannot be changed, except for the expiration of accepted credentials. This ensures the completion of the delivery within a finite amount of time. Moreover, it is crucial to consider that the blockchain timestamps all transactions performed on it.
- **Availability**. The digital credential is always accessible and usable.
  With digital credentials, individuals no longer need paper certificates or worry about misplacing documents. These credentials can be securely stored and accessed on digital platforms. Unlike schemes based on central servers, blockchain approaches ensure that all data necessary for verifying digital credentials is available to all participants at all times. In our proposal, the digital credential is stored in the blockchain in the form of a Rejectable SoulBound Token, associated with the wallet of the corresponding holder. Since this information is stored on a distributed system, it is always accessible and ready to use by its holder.
- **Fairness and transferability of evidence**. The issuance of the digital credential is a fair operation, and the generated evidence is transferable to external users.

Users must use digital certificates in a way that is fair. This means issuers, holders, and verifiers must refrain from processing the data in the digital certificates in a manner that is detrimental or misleading to the individuals concerned. Our proposal has considered how the digital certificates processing may affect the individuals concerned and how it can guarantee fairness. During the execution of the protocol, the issuer and the holder exchange the digital credentials as evidence of the acceptance of the T&C. This exchange must be fair and the parties must be capable of presenting or transmitting evidence associated with the issued digital credential to relevant parties, during authentication or authorization processes. The introduction of the blockchain technology and the definition of RejSBTs reassure all parties that any disputes or issues will be handled with transparency and fairness.

The fairness of the proposed protocol for digital credentials management can be demonstrated. At the end of a protocol execution, the holder has possession of the token (RejSBT token) that represents the digital credential in her wallet and, at the same time, I can prove that H has accepted the T&C associated with the digital credential. Moreover, each party has either received the proper elements (token and acceptance of T&C evidence), or neither party has received any useful data about the other's element, providing *strong fairness* [24]. This property is achieved due to the fact that the protocol uses the same function of the smart contract, `acceptTransfer()`, to accept the T&C and transfer the token. Since the functions `acceptTransfer()` and `mint()` validate the identity of the executors and the transactions are signed operations, it is proved that the right parties have executed the functions. Moreover, the evidence generated by the protocol can be verified by an external party in order to prove the outcome and the effects of the exchange, since the transactions are registered on the blockchain.

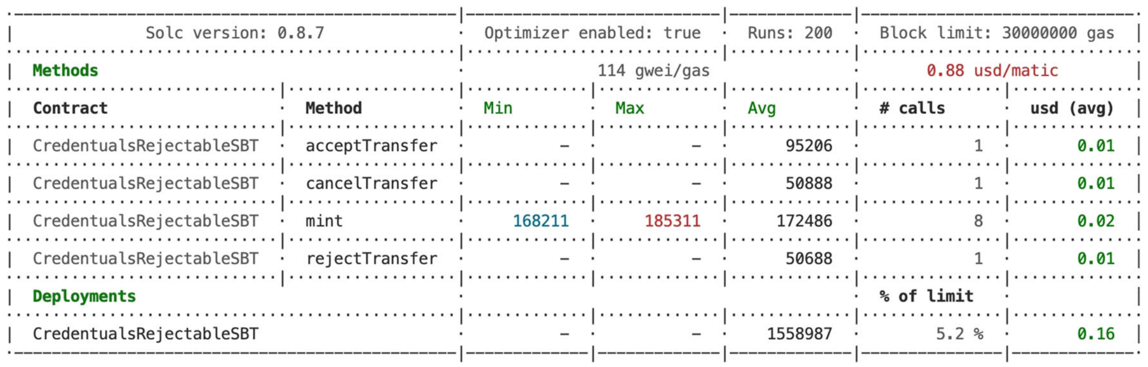
- **Non-repudiation**. The issuer of the digital credential cannot deny having issued the credentials. Moreover, the digital credential holder cannot deny their reception. That is, the digital credential management protocol provides Non-Repudiation of Reception (NRR) evidence for the credentials and Non-repudiation of Origin (NRO) evidence too. Additionally, the system provides Non-Repudiation of the acceptance of T&C. This way, the digital credential holder cannot deny having accepted them.

  - Regarding the non-repudiation of origin evidence, I cannot deny having executed the (*Issuance of the digital credential*) step to create the digital credential. This is because there is a *mint* of a token by his address containing the token identifier, the deadline, and the identity of the digital credential holder in the `_transferableOwners` mapping. The related smart contract can prove that the *Receiver acceptance* step has been executed and that the final state of this credential delivery is *Accepted*, demonstrating that H has received the token containing the credentials data in its metadata.
  - Concerning the non-repudiation of reception evidence, H cannot deny the reception of the credentials because an `acceptTransfer()` transaction from her address is stored on the blockchain. This transaction signifies the acceptance of the credentials' reception, and accordingly, the smart contract changed the state of the notification to *Accepted* after transferring the token from I to H.
  - Regarding the non-repudiation of the acceptance of T&C associated with the credential, H cannot deny having accepted it because these T&C are related to the acceptance of the token. We have presented two alternatives for storing the T&C. The first is to include them in the metadata associated with the token. The other one involves the use of an external storage system, such as IPFS. In this case, I uploads both the credentials and the T&C to a decentralized storage system and then includes the identifier of the data in the token. In both cases, since the information included in the token is immutable, H's credentials are strongly linked to these terms.

- **Selective reception**. The protocol allows the holder to decide whether to accept or decline the reception of the digital credential. In the case of digital credentials, any holder must have the capability of ignoring or discounting a credential created by any issuer that is inconsistent with the recipient's attitudes, opinions, or beliefs.
  Thanks to the use of rejectable soulbound tokens, H has the ability to choose between accepting or rejecting the digital credential. This property was not achievable with the standard versions of the tokens, NFT and SBT. Thanks to the definition of the new kind of tokens, holders can avoid the association of their identity with credentials they prefer not to have, preventing the transfer of tokens to their wallet.

7 Performance analysis

With the implementation of this protocol, we have tested its performance using the Hardhat development environment. Hardhat is an Ethereum development environment that facilitates the calculation of gas usage per unit test by providing metrics for the executed functions and deployments.



```
·-----------------------------------------------|---------------------------|-------------|----------------------------·
|              Solc version: 0.8.7              ·  Optimizer enabled: true  ·  Runs: 200  ·  Block limit: 30000000 gas  │
················································|···························|·············|·····························
|  Methods                                      ·        114 gwei/gas       ·              0.88 usd/matic              │
·····························|··················|·············|·············|·············|···············|·············
|  Contract                  ·  Method          ·  Min        ·  Max        ·  Avg        ·  # calls      ·  usd (avg)  │
·····························|··················|·············|·············|·············|···············|·············
|  CredentualsRejectableSBT  ·  acceptTransfer  ·          -  ·          -  ·      95206  ·            1  ·       0.01  │
·····························|··················|·············|·············|·············|···············|·············
|  CredentualsRejectableSBT  ·  cancelTransfer  ·          -  ·          -  ·      50888  ·            1  ·       0.01  │
·····························|··················|·············|·············|·············|···············|·············
|  CredentualsRejectableSBT  ·  mint            ·     168211  ·     185311  ·     172486  ·            8  ·       0.02  │
·····························|··················|·············|·············|·············|···············|·············
|  CredentualsRejectableSBT  ·  rejectTransfer  ·          -  ·          -  ·      50688  ·            1  ·       0.01  │
·····························|··················|·············|·············|·············|···············|·············
|  Deployments                                  ·                                         ·  % of limit   ·             │
················································|·············|·············|·············|···············|·············
|  CredentualsRejectableSBT                     ·          -  ·          -  ·    1558987  ·        5.2 %  ·       0.16  │
·-----------------------------------------------|-------------|-------------|-------------|---------------|-------------·
```

Fig. 2 Table of costs of the implemented protocol

To ensure the accuracy of the protocol, we conducted several unit tests and evaluated their gas cost efficiency. Specifically, we have configured Hardhat to utilize a fork of the Polygon PoS network for testing purposes. Figure 2 presents the data obtained from the Hardhat environment. It is important to note, as depicted in Fig. 2, that during the execution of the tests (November 2023), the price of MATIC was 0.88 USD, and the corresponding gas price was 114 gwei.

The data provided by Hardhat demonstrates the successful execution of all the functions that comprise the digital access credentials protocol implementation.

Analyzing the Hardhat gas cost results, the most expensive function is the smart contract deployment, which costs 0.16 USD. All the other functions, when compared to the deployment, are really inexpensive. The `mint()` function, responsible for issuing the new digital credential, has the highest price of 0.02 USD. The `acceptTransfer()` function, involving the change of ownership, is slightly more expensive than the `cancelTransfer()` and `rejectTransfer()` functions, which only remove the proposed transferable owner.

In addition to the obtained Hardhat results, to visually represent the cost trends over the past year, we have applied the average gas cost data of the Polygon PoS network from the last year to the Hardhat results. These results are presented in the graphs of Figs. 3 and 4. The first graph shows the USD

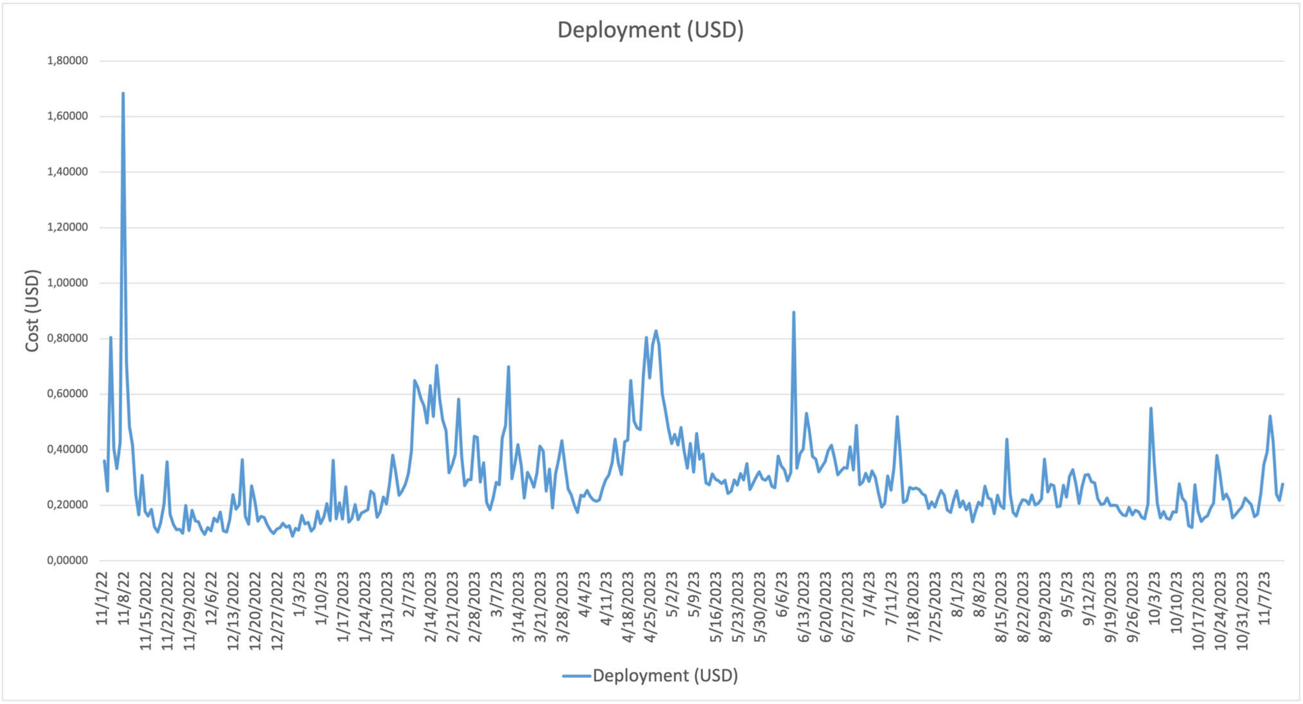


Fig. 3 Estimated average cost of the deployment of the smart contract (measured in USD) over the past year

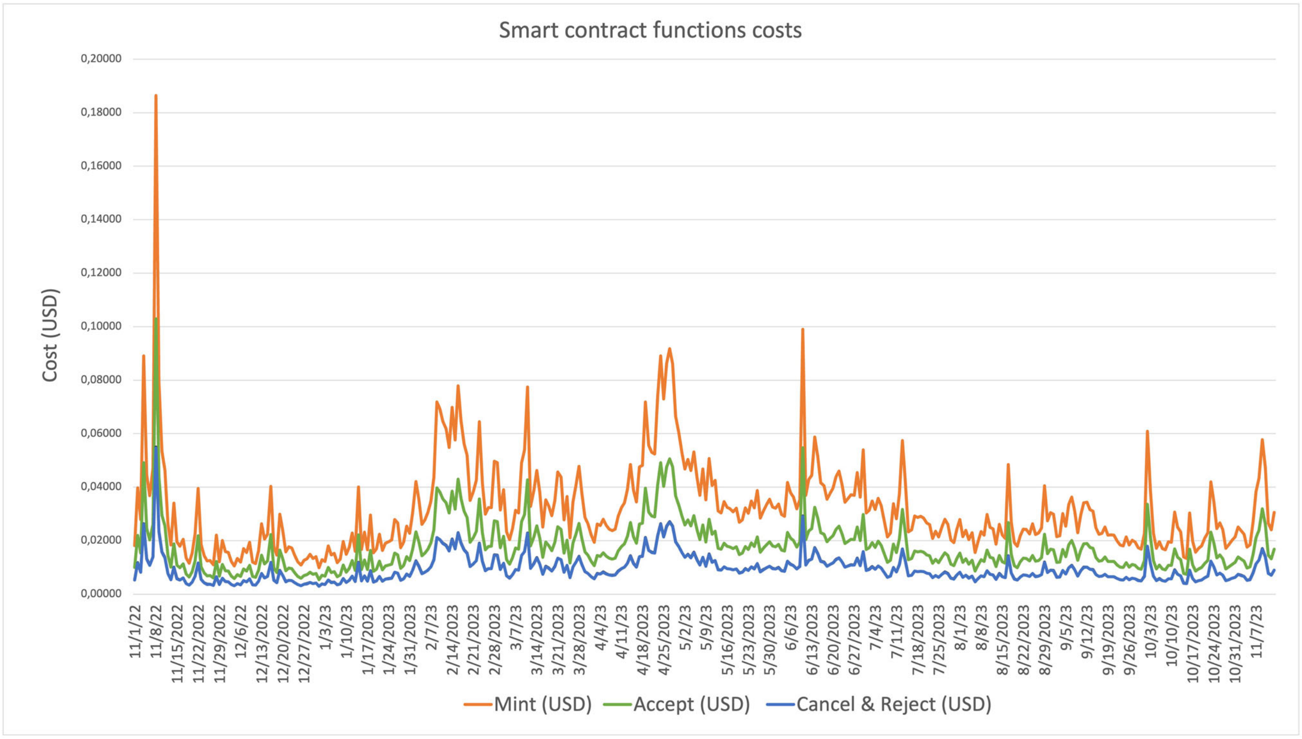


Fig. 4 Estimated average cost of the different functions defined in the smart contract (measured in USD) over the past year

costs of the smart contract deployment, while the second represents the costs of the execution of the different functions of the protocol. It's important to note that in the graph, we have combined the functions `cancelTransfer()` and `rejectTransfer()` because these two functions have very similar cost results, and their separate representation on the graph would result in overlap.

As previously discussed in the analysis of the Hardhat cost results, the average representation of results follows the same pattern. Specifically, with the data obtained last year, the smart contract deployment has an average cost of 0.28660 USD, as shown in Table 1. The average costs of executing the different functions range from 0.032 USD for the `mint()` function to 0.018 USD for the `acceptTransfer()` function and further down to 0.009 USD for the `cancelTransfer()` and `rejectTransfer()` functions.

Table 1 Average transactions costs: November 2022–November 2023

	US Dollars
CredentialsRejectableSBT deployment	0.28660
AcceptTransfer	0.01750
CancelTransfer	0.00936
Mint	0.03171
RejectTransfer	0.0.00932

8 Conclusions

The paper presents a novel proposal to combine the use of digital tokens with credential management systems. The result is a powerful solution that manages the credentials and at the same time handles the acceptance of the T&C related to the use of these credentials. Moreover, it generates non-repudiation evidence of both the reception of the credentials and the acceptance of the terms.

Since blockchain offers some inherent features that are highly advantageous for the management of credentials, tokens are an interesting element to represent credentials. However, current token standards do not fulfill the desired properties for credentials, such as non-transferability along with rejectability. In response, we introduce a new kind of token, the RejSBTs, to represent credentials and associated terms, providing non-repudiation of reception and origin proofs and achieving the aforementioned properties of non-transferability and rejectability.

Our proposal results in a digital access credential protocol that ensures interoperability, allowing the usage of the same digital access credential across multiple platforms. At the same time, it empowers individuals with self-sovereign identity, placing the holder as the primary responsible party for the token.

The proposed protocol has been implemented and evaluated to prove its viability, showing that it is an interesting system for the management of credentials.

It's essential to acknowledge that under certain circumstances, such as after a verification process or due to the behavior of the holder, a credential may require revocation. In such cases, the issuer can claim to a trusted third party (e.g., a judge in a court of law) to resolve the conflict. Additionally, holders may wish to disassociate themselves from the digital credential, effectively unsubscribing their identity from the system. Consequently, as part of future work, we aim to explore various possibilities and methods to adapt the proposed protocol, facilitating the revocation and cancellation of digital credentials while ensuring robust security and safeguarding user privacy.

Acknowledgements Grant PID2021-122394OB-I00 (BLOBSEC) funded by MICIU/AEI/ 10.13039/501100011033 and by ERDF/EU.

Author contribution Rosa Pericás-Gornals, Maciá Mut-Puigserver and M. Magdalena Payeras-Capellá wrote the main manuscript text. Jaume Ramís-Bibiloni did the state of art. Miquel Á. Cabot-Nadal and Rosa Pericàs Gornals wrote the implementation. Maciá Mut-Puigserver and M. Magdalena Payeras-Capellá wrote the security analysis. Jaume Ramís-Bibiloni, Rosa Pericás Gornals and Miquel Á. Cabot-Nadal did the performance analysis. All authors reviewed the manuscript

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data availability No datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Brands D (2002) A technical overview of digital credentials
2. Weyl EG, Ohlhaver P, Buterin V (2022) Decentralized society: finding web3's soul. https://doi.org/10.2139/ssrn.4105763
3. Cabot-Nadal MA, Payeras-Capellá M, Mut-Puigserver M, Soto-Fernández A (2022) Improving the token ERC-721 implementation for selective receipt: rejectable NFTs, In: 2022 6th International conference on system reliability and safety (ICSRS), pp 243–250. https://doi.org/10.1109/ICSRS56243.2022.10067494
4. Pericás-Gornals R, Mut-Puigserver M, Payeras-Capellá MM, Cabot-Nadal MÁ, Ramis-Bibiloni J (2023) Rejectable soulbound tokens for credentials assignment and acceptance of terms. In: 2023 7th Cyber security in networking conference (CSNet), pp 212–218. https://doi.org/10.1109/CSNet59123.2023.10339728
5. Brands S (1997) Privacy-protected transfer of electronic information. U.S. patent ser. no. 5,604,805. Filed August 1993
6. Grassi P, Garcia M, Fenton J (2020) Digital identity guidelines. NIST Special Publication 800:63–3
7. Allen C (2002) The path to self-sovereign identity. Life with alacrity. https://www.lifewithalacrity.com/article/the-path-to-self-sovereign-identity/
8. Agarkar A, Karyakarte M, Chavhan G, Patil M, Talware R, Kulkarni L (2024) Blockchain aware decentralized identity management and access control system. Measurement: Sensors 3(1):101032. https://doi.org/10.1016/j.measen.2024.101032
9. Du Z, Li Y, Fu Y, Zheng X (2024) Blockchain-based access control architecture for multi-domain environments. Pervasive Mob Comput 98:101878. https://doi.org/10.1016/j.pmcj.2024.101878
10. Cabot-Nadal M, Playford B, Payeras-Capellá M, Gerske S, Mut-Puigserver M, Pericás-Gornals R (2023) Private identity-related attribute verification protocol using soulbound tokens and zero-knowledge proofs, In: 2023 7th Cyber Security in Networking Conference (CSNet), pp 153–156. https://doi.org/10.1109/CSNet59123.2023.10339754
11. Mecozzi R, Perrone G, Anelli D, Saitto N, Paggi E, Mancini D (2022) Blockchain-related identity and access management challenges: (de)centralized digital identities regulation, in. IEEE International Conference on Blockchain (Blockchain) 2022:443–448. https://doi.org/10.1109/Blockchain55522.2022.00068
12. Herbke P, Yildiz H (2022) ELMO2EDS: transforming educational credentials into self-sovereign identity paradigm. In: 20th International conference on information technology based higher education and training (ITHET), pp 1–7. https://doi.org/10.1109/ITHET56107.2022.10031276
13. Reza M, Biswas S, Alghamdi A, Alrizq M, Bairagi AK, Masud M (2021) ACC: blockchain based trusted management of academic credentials, in. IEEE International Symposium on Smart Electronic Systems (iSES) 2021:438–443. https://doi.org/10.1109/iSES52644.2021.00104
14. Eltuhami M, Abdullah M, Talip B (2022) Verification identity traceability document in digital identity systems using non-transferable non-fungible tokens. In: 2022 International visualization. information technology conference (IVIT), pp 136–142. https://doi.org/10.1109/IVIT55443.2022.10033362
15. Hunhevicz J, Bucher D, Soman RK, Honic M, Hall D, De Wolf C (2023) Web3-based role and token data access: the case of building material passports. In: European conference on computing in construction, 40th international CIB W78 conference.https://doi.org/10.35490/EC3.2023.217
16. BCdiploma. WEB3 digital credentials in academics: everything you need to know to get started. https://www.bcdiploma.com/en/blog/digital-credentials-web3
17. Chartrand Jea. Building the digital credential infrastructure for the future. Digital Credentials Consortium. https://digitalcredentials.mit.edu/docs/white-paper-building-digital-credential-infrastructure-future.pdf
18. Saramago RQ, Meling H, Jehl LN (2023) A privacy-preserving and transparent certification system for digital credentials. In:

26th International conference on principles of distributed systems (OPODIS 2022). Schloss Dagstuhl-Leibniz-Zentrum für Informatik. https://doi.org/10.4230/LIPIcs.OPODIS.2022.9

19. Gerbershagen D. Analysis of the state of the art and the practice of digital credentialing. https://wwwmatthes.in.tum.de/file/16b2rretkx0f2/Sebis-Public-Website/-/Master-s-Thesis-Dominik-Gerbershagen/200311_Gerbershagen_MA_Thesis.pdf
20. Fang J, Feng T, Guo X, Ma R, Lu Y (2024) Blockchain-cloud privacy-enhanced distributed industrial data trading based on verifiable credentials. Journal of Cloud Computing 13. https://doi.org/10.1186/s13677-023-00530-7
21. Puigserver M, Payeras-Capellá M, Ferrer-Gomila J, Vives Guasch A, Castellá-Roca J (2012) A survey of electronic ticketing applied to transport. Computers & Security 31:925–939. https://doi.org/10.1016/j.cose.2012.07.004
22. Grassi P, Fenton J, Lefkovitz N, Danker J, Choong Y-Y, Greene K, Theofanos M (2017) NIST SP 800-63A - digital identity guidelines: enrollment and identity proofing https://doi.org/10.6028/NIST.SP.800-63a
23. Verifiable Credentials Data Model v1.1 W3C Recommendation 03 March 2022, w3c.https://www.w3.org/TR/vc-data-model/
24. Asokan N, Schunter M, Waidner M (1997) Optimistic protocols for fair exchange. In: Proceedings of the 4th ACM conference on computer and communications security, pp 7–17. https://doi.org/10.1145/266420.266426