mathematics

Article

Digital Authentication System in Avatar Using DID and SBT

Geunyoung Kim and Jaecheol Ryou *

Department of Computer Science and Engineering, Chungnam National University, Daejeon 34134, Republic of Korea; gykim@cnu.ac.kr
* Correspondence: jcryou@cnu.ac.kr

Abstract: Anonymity forms the basis of decentralized ecosystems, leading to an increase in criminal activities such as money laundering and illegal currency trading. Especially in blockchain-based metaverse services, activities such as preventing sexual crimes and verifying the identity of adults are becoming essential. Therefore, avatar authentication and the KYC (Know Your Customer) process have become crucial elements. This paper proposes a mechanism to achieve the KYC process by verifying user identity using smart contracts. Users obtain an SBT (Soul Bound Token) from the metaverse service provider through the DID (Decentralized Identity) credential issued during the KYC process. The identity verification of avatars occurs within smart contracts, ensuring user privacy and protection through ZKP (Zero Knowledge Proof). Tools for generating ZKP are also provided, enabling users, even those who are unfamiliar with ZKP, to use them conveniently. Additionally, an integrated wallet is offered to seamlessly manage DID credentials and SBTs. Furthermore, in case of avatar identity issues, users can request an audit by the issuer through the associated DID tokens.

Keywords: blockchain; digital identity; smart contract; avatar; Soul Bound Token; KYC

MSC: 68M14; 68-04

Citation: Kim, G.; Ryou, J. Digital Authentication System in Avatar Using DID and SBT. *Mathematics* **2023**, *11*, 4387. https://doi.org/10.3390/math11204387

Academic Editor: Jan Lansky

Received: 11 September 2023
Revised: 15 October 2023
Accepted: 20 October 2023
Published: 22 October 2023

1. Introduction

The emergence of cryptocurrency using blockchain technology provides decentralization and innovative opportunities for the financial system, but at the same time, it causes many problems. In particular, it is raising concerns about the safety and regulation of the financial system. The most serious problems are AML (Anti-Money Laundering) and CFT (Counter-Financing of Terrorism). According to Chainalysis, USD 23.8 billion worth of cryptocurrency, a 68% increase compared to the previous year, was laundered in 2022 [1]. In addition, due to the borderless nature of cryptocurrency, unlimited foreign exchange transactions are possible and lead to capital flight, which can affect the sustainability of the national economy [2]. Due to the anonymity provided by cryptocurrency, it is not easy to verify the real identity of cryptocurrency users. As a result, concerns about the reliability and safety of financial transactions using cryptocurrency are being raised, and user authentication and KYC (Know Your Customer) processes are essential to solve this problem.

In particular, user authentication is becoming an increasingly vital element even in the metaverse environment that epitomizes Web3, operating through blockchain technology. In this case, there is a drawback of managing all users' personal information on the server, which includes issues such as manipulating game item acquisition. Indeed, in May 2020, there was an incident where the personal information of 100 million active users of Roblox was compromised due to a hacking incident [3]. However, in the case of decentralized metaverse systems (e.g., Decentraland [4] and Sandbox [5]) that utilize blockchain, users log in using crypto wallet software (e.g., Metamask (version 11.2.0)) to map with avatars in the metaverse, and provide services through the ownership of NFTs present in their wallets. This is because crypto wallet software relies on verifying the possession of specific items rather than issuing NFTs based on the user's identity, making it an inadequate method for identity authentication. Even if an NFT authenticated the user, the information could still be publicly visible on the blockchain, potentially leading to privacy issues.

User authentication is essential not only in metaverse environments but also in the majority of Web3 applications. In the field of finance, centralized exchanges already implement KYC procedures to verify users' identities. Moreover, regulations like the Travel Rule [6] for anti-money laundering efforts require VASPs (Virtual Asset Service Providers) to collect and transmit information, such as the sender's and receiver's names, account numbers, addresses, and more. This mandatory reporting helps to mitigate the risks associated with money laundering and terrorist financing. Therefore, the FATF (Financial Action Task Force) emphasizes the importance of identity authentication to the extent that, through its first set of VASPs guidelines released in June 2019, it recommends that countries advocate for the enforcement of the Travel Rule. Similarly, decentralized exchanges also make various efforts to verify users' identities. Among them, tbdex [7] is developing a decentralized protocol for exchanging fiat currency and cryptocurrencies. They are utilizing DID (Decentralized Identity) credentials [8] to authenticate users' identities as part of their protocol development process.

As such, it is important for identity authentication to determine the identity of the user associated with the wallet address. Vitalik Buterin, the creator of Ethereum, proposed the concept of the SBT (Soul Bound Token) [9], which issues non-transferable NFTs to represent users' identities, offering a proposal for identity verification and validation within decentralized societies. The world's largest cryptocurrency exchange, Binance, has issued a form of identity token called the BABT (Binance Account Bound Token) [10], a form of SBTs. These tokens are issued to users who have completed Binance's KYC process, enabling them to participate in various projects based on the BNB chain and receive rewards accordingly. On the other hand, in order to proceed with user identity management using the SBT, it is essential to verify that the actual wallet user wishes to be issued an SBT.

In the blockchain ecosystem, it is common to verify a user's identity using DID and an SBT. However, DID verification needs to be conducted in the off-chain environment outside of the blockchain, and users have to submit credentials every time they want to prove their identity, which is a drawback. On the other hand, using an SBT, identity verification can occur on-chain within the blockchain environment based on the possession of an SBT. However, to issue an SBT, users need to provide their private information to the SBT issuer, raising concerns about privacy protection.

In this paper, we propose a scheme that utilizes DID, which is globally adopted under the W3C initiative, to verify users through smart contracts and then issue an SBT, enabling a seamless integration in the Web3 environment, especially in the metaverse. To protect user privacy, instead of transmitting the DID as it is, the verification requires a VASP, offering services using proof based on a ZKP system, ensuring that critical user information remains unknown. Furthermore, users can generate cryptographic proof through tools designed for complex ZKP without having to create them directly, increasing user convenience. Additionally, we present a wallet that manages DID credentials and SBTs as a unified seed value, thereby enhancing user convenience.

The rest of the paper is organized as follows: first, Section 2 explains the technologies employed in this paper; Section 3 proposes the scheme; Section 4 analyses the privacy and security aspects of the proposed scheme; Section 5 reviews related work; Section 6 presents discussions; and, finally, our conclusions are presented in Section 7.

2. Background

2.1. DID

DID is an identity information system used within a trust-distributed framework without centralized registration mechanisms. It is a technology based on distributed ledgers that enable individuals to autonomously manage proof of identity and control the scope and recipients of identity information submissions.

As shown in Figure 1, the part that represents user information is called a claim. These claims are collected, signed by the issuer, and sent to the holder in the form of credentials. The holder then stores the credentials and, when they want to reveal specific information, they add the information and their own signature and send it to the verifier, allowing the user's information to be verified. For example, if an accommodation facility requires age verification, a national ID card presented to staff will reveal all personal information, such as address, social security number, and name, not just date of birth. However, by using DID, it is possible to prove something like "19 years or older" without revealing the exact age. DID technology has gained significant attention because it uses the principles of Self-Sovereign Identity (SSI), which allows users to store and manage authentication data while providing only the information necessary for identity verification [11]. In the medical field, methods for face-to-face consultations and personal health devices (PHDs) have been proposed for individual health management [12,13]. As these online services require user authentication, research using DID for personal information protection is being conducted to address issues related to sensitive medical information in electronic health records [12]. In the era of data economy, DID technology is widely used in many authentication services for reasons such as data sovereignty and personal information protection. However, there is a drawback in using DID in Web3 systems.

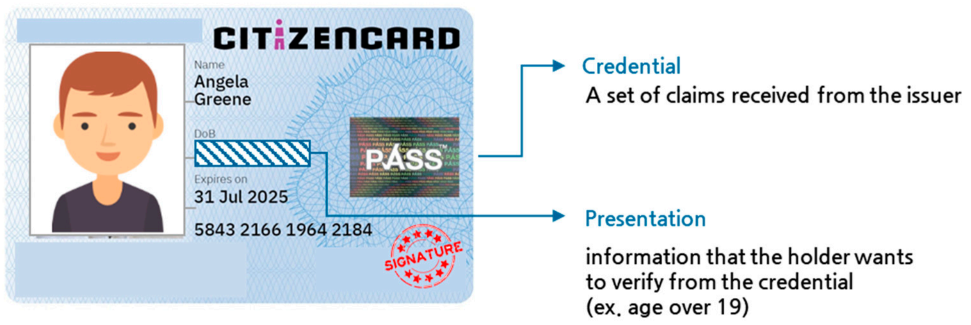
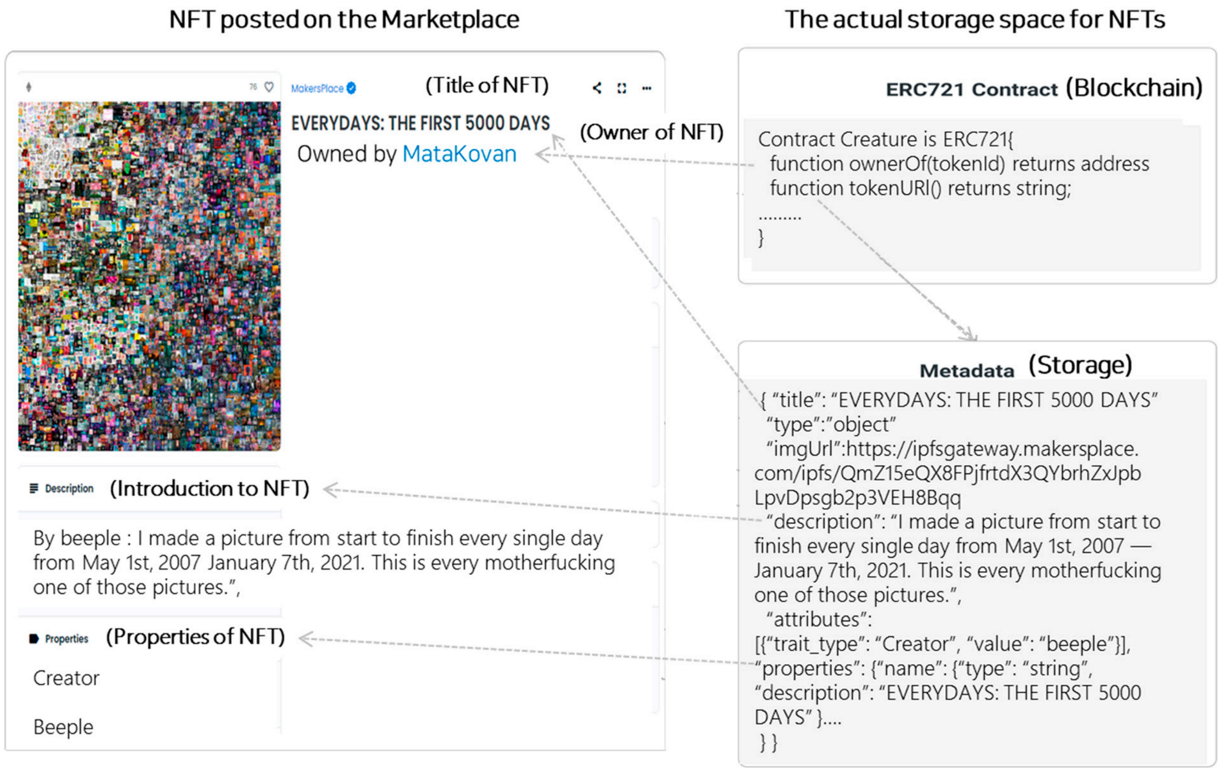

CITIZENCARD
Name
Angela Greene
DoB
Expires on
31 Jul 2025
5843 2166 1964 2184
PASS
SIGNATURE
Credential
A set of claims received from the issuer
Presentation
information that the holder wants to verify from the credential (ex. age over 19)

Figure 1. Components of DID.

2.2. NFT

NFT stands for "Non-Fungible Token", representing a unique and irreplaceable token. Each NFT has a distinct identifier, which encapsulates its unique attributes and ownership information. This data is uploaded to the blockchain, making forgery or tampering difficult. NFTs grew based on the ERC-721 [14] standard in Ethereum [15]. Ethereum is a decentralized platform built on blockchain technology, best known for its smart contract capabilities [16]. Smart contracts are encoded segments of the EVM (Ethereum Virtual Machine) bytecode, which is a Turing-complete programming language.

Within the blockchain's on-chain environment, there is an "ownerOf" function that indicates the owner's address and a "tokenURI" function that contains a URL to a JSON file with various attributes, including the image of the NFT. The actual information of NFTs is stored off-chain, either on IPFS or a server. Utilizing this feature, verifying the ownership and data information of a specific NFT by using its ID (tokenID) on the blockchain is possible. As shown in Figure 2, NFT marketplaces provide a user interface that allows for easy viewing of NFT information based on the data uploaded to the blockchain. NFTs are primarily used to represent ownership of digital content, such as digital art, music, videos, and game items. Therefore, it provides the advantage of protecting the owner's copyright from the work and transparently managing the rights of digital assets. Currently, blockchain-based metaverse services that support NFT transactions operate as shown in Figure 3.

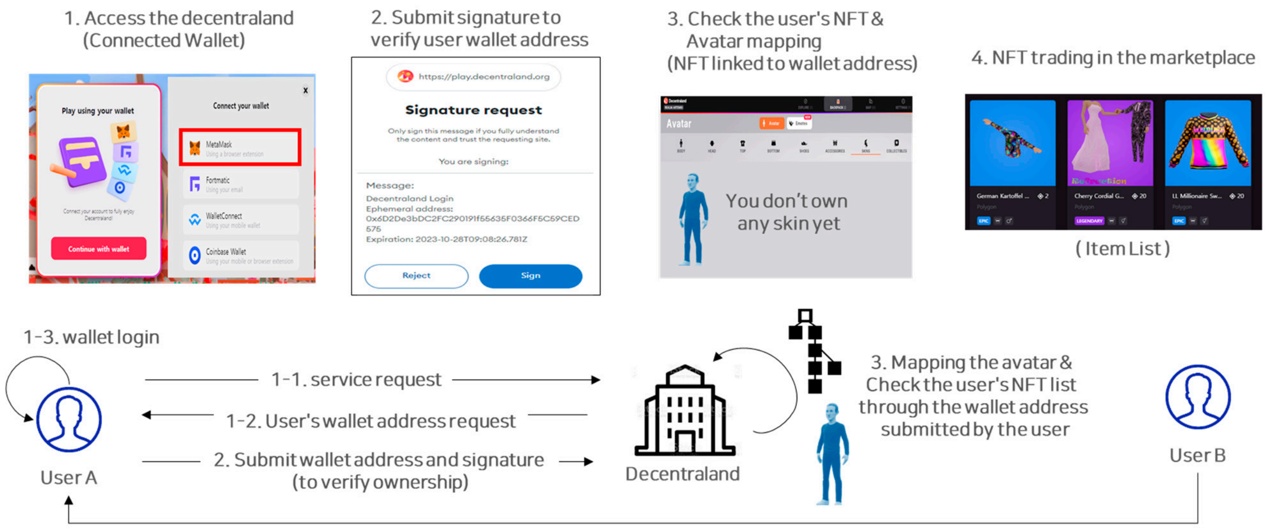

NFT posted on the Marketplace
The actual storage space for NFTs
MakersPlace
(Title of NFT)
EVERYDAYS: THE FIRST 5000 DAYS
Owned by MataKovan
(Owner of NFT)
Description (Introduction to NFT)
Properties (Properties of NFT)
Creator
Beeple
ERC721 Contract (Blockchain)
Metadata (Storage)

By beeple : I made a picture from start to finish every single day from May 1st, 2007 January 7th, 2021. This is every motherfucking one of those pictures.",

```
Contract Creature is ERC721{
  function ownerOf(tokenId) returns address
  function tokenURI() returns string;
  .........
}
```

```
{ "title": "EVERYDAYS: THE FIRST 5000 DAYS"
  "type":"object"
  "imgUrl":https://ipfsgateway.makersplace.
com/ipfs/QmZ15eQX8FPjfrtdX3QYbrhZxJpb
LpvDpsgb2p3VEH8Bqq
  "description": "I made a picture from start to
finish every single day from May 1st, 2007 —
January 7th, 2021. This is every motherfucking
one of those pictures.",
  "attributes":
[{"trait_type": "Creator", "value": "beeple"}],
"properties": {"name": {"type": "string",
"description": "EVERYDAYS: THE FIRST 5000
DAYS" }....
 } }
```

Figure 2. Composition of NFT.


1. Access the decentraland (Connected Wallet)
Play using your wallet
Continue with wallet
Connect your wallet
MetaMask
Fortmatic
WalletConnect
Coinbase Wallet
2. Submit signature to verify user wallet address
https://play.decentraland.org
Signature request
You are signing:
Reject
Sign
3. Check the user's NFT & Avatar mapping (NFT linked to wallet address)
Avatar
You don't own any skin yet
4. NFT trading in the marketplace
(Item List)
1-3. wallet login
1-1. service request
1-2. User's wallet address request
2. Submit wallet address and signature (to verify ownership)
User A
Decentraland
3. Mapping the avatar & Check the user's NFT list through the wallet address submitted by the user
User B

Message:
Decentraland Login
Ephemeral address:
0x6D2De3bDC2FC29019f55635F0366F5C59CED
575
Expiration: 2023-10-28T09:08:26.781Z

Figure 3. Login procedure in metaverse service using digital wallet.

First, the user must log in to a digital wallet program such as Metamask and go through the process of verifying the signature value to verify the user's wallet address. After that, the wallet address and the corresponding avatar are mapped 1:1. In this way, the blockchain wallet address supported by the metaverse can be expanded into an ID concept that allows avatars to log in. In addition, in Decentraland, users who are 19 years of age or older need a costume of an ice breaker character issued as an NFT to participate in hold'em poker. User authentication is carried out whether or not a specific NFT is possessed. However, NFTs are unsuitable for creating user identities as NFTs because the owner can transfer ownership.

2.3. SBT

In May 2022, the co-founders of Ethereum, Vitalik Buterin, economist Glen Weyl, and lawyer Puja Ohlhaver, published a paper titled "Decentralized Societies: Finding the Soul of Web3.0", which introduced the concept of an SBT. The paper proposed the development of a new type of cryptographic token called a "Soul Bound Token". These tokens are designed to represent individual identities in a decentralized digital world. It introduced the concept that these tokens could serve as the foundation for creating a decentralized society (DeSoc) like the metaverse by making it impossible to transfer or trade them, unlike the traditional NFT. Utilizing an SBT, various aspects of an individual's identity, such as employment, volunteer work, educational history, qualifications, medical records, criminal records, membership status, and affiliations, can be securely established by creating non-transferable NFT badge forms. These badges are then mapped to the user's digital wallet address, ensuring the preservation of their identity. Currently, the SBT is creating various ERC standards (ERC-5192 [17], ERC-5727 [18], and ERC-6454 [19]). Generally, the minimum functions and interfaces with non-transferable characteristics are being defined.

Many services are attempting to implement identity verification systems using the SBT. In [20], a Dapp was developed to issue and verify COVID-19 vaccination certificates in the form of an SBT. In [21], a Dapp was developed to provide digital certificates in the education sector using an SBT. In [22], various cases of SBT use in the Web3 domain were listed, confirming its potential to replace the existing ownership-guaranteeing NFTs. Thus, the emergence of the SBT has led to significant advances in user identity verification systems within the blockchain.

2.4. ZKP

ZKP is a crucial cryptographic concept that provides a means for a prover to demonstrate the truth of a statement to a verifier without divulging any details about the statement, except for its validity. This cryptographic process entails a series of interactive exchanges between the prover and verifier. During these interactions, the prover seeks to authenticate the statement by sharing specific information while preserving the confidentiality and privacy of the statement itself. Through these exchanges, the verifier can securely confirm the veracity of the prover's claim without compromising the underlying information. A typical ZKP framework comprises three distinct phases [23]:

- Witness Phase: in the initial stage, the prover formulates a proof based on the statement and transmits it to the verifier.
- Challenge Phase: subsequently, the verifier formulates a challenge based on the proof received and forwards it to the prover.
- Response Phase: finally, the prover tackles the challenge and sends the response back to the verifier.

ZKP is a fundamental tool in modern cryptography, gaining significant attention due to its ability to address critical security and privacy challenges. It finds application in various domains, including digital identity [24], blind signatures [25], and attribute verification [26]. ZKP offers a potent cryptographic technique that enhances privacy, security, and efficiency in blockchain applications. There are numerous applications for implementing ZKP in the blockchain context, such as enabling anonymous transactions [27], facilitating secure voting systems [28,29], and improving supply chain management [30].

3. Identity Authentication Scheme in Metaverse

In current metaverse systems, for KYC procedures, metaverse service providers handle user information separately to verify identities. When granting specific permissions based on verified identities, centralized methods are used to manage user privileges. In this scheme, metaverse service providers need to handle user information, and users have to transmit their information to each metaverse service provider, leading to potential hacking risks associated with centralized databases. In cases where a system issues an SBT without relying on a centralized database to authenticate user identities, the verification must occur transparently through the blockchain, rather than the service provider confirming the user's identity. While user information can be protected through ZKP or encryption, it is challenging to ascertain whether the user has provided accurate information. To address this challenge, we aim to use the globally adopted DID to prove that the provided user information comes from a trustworthy issuer. Moreover, once the user possesses an identity-related SBT, they only need to prove possession of an SBT in their wallet when accessing different metaverse services. Consequently, there is no requirement to transmit credentials or identity information to each distinct metaverse service provider. Detailed information regarding the SBT, as demonstrated in Section 3.5, does not include users' personal data. Only the issuer managing the DID credentials can access this information. Therefore, if issues arise in the future, the service provider can request DID information to manage the user's privileges effectively. This approach enhances privacy and security in metaverse systems.

3.1. System Architecture

The proposed scheme consists of four entities as depicted in Figure 4: user, issuer, SBT issuer, and verifier. The user maps with the metaverse avatar through the wallet address that has the same seed value as the wallet that manages the DID credential. The SBT issuer is a government department that manages users' identities, handles personal credentials, and is responsible for issuing an SBT in the metaverse, DeFi, or DEX. Lastly, the verifier must utilize smart contracts to grant service permissions to users who possess a specific SBT.

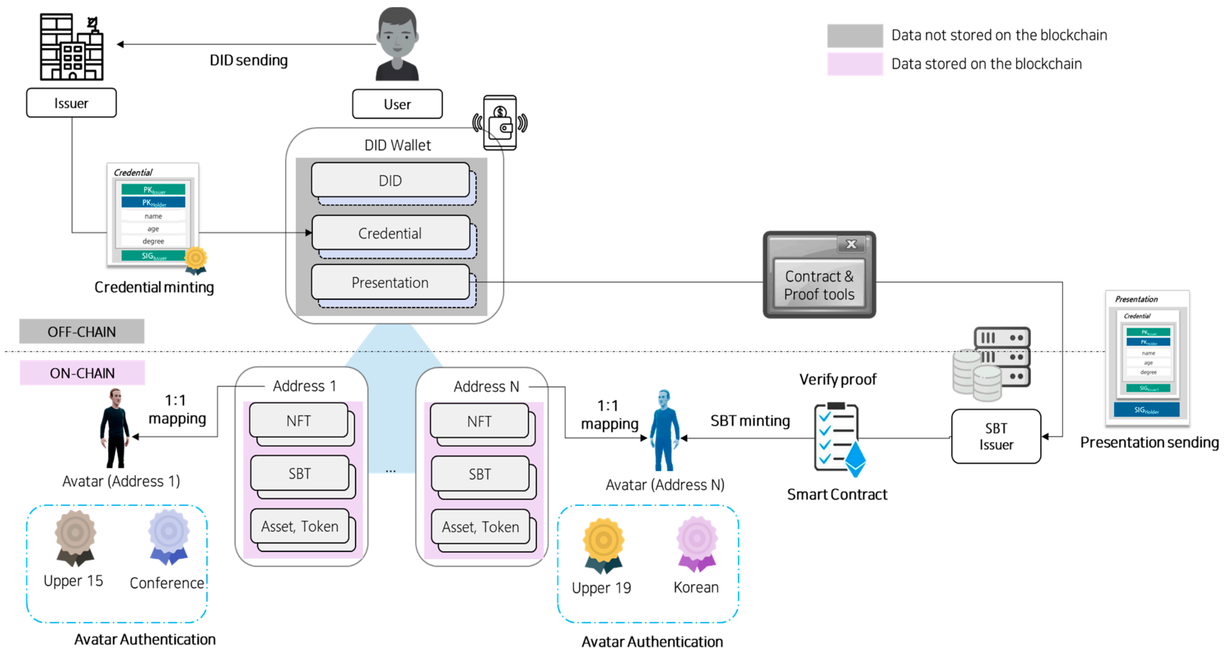


Figure 4. Diagram of avatar identity authentication system in metaverse combining DID and SBT.

3.2. Program Flow

(1) Users provide their DID to the credential issuer and submit the information required for KYC processes, similar to providing information for an ID card. Once the user's information is verified, the issuer delivers a credential that can demonstrate qualifications and identity, similar to a form of identification.

(2) Users store the credential in their wallet.

(3) The service provider creates a smart contract to verify and generate the SBT with specific permissions, then deploys this contract onto the Ethereum network.

(4) Users create a presentation by adding their signature to the credential and generating proof that fulfills the conditions established in step (3).

(5) The smart contract verifies the proof provided by the user and, if the conditions are met, issues the SBT to the user's wallet address.
(6) Users log in to their avatars using the wallet address where the SBTs are stored.
(7) In a specific service, permission for usage is granted based on the presence of an SBT.

3.3. *Credential/Presentation*

DID is used to verify the user's identity, as shown in Figure 5. Based on the user's Decentralized Identifier, the issuer creates a credential. During this process, the user's personal information used for authentication is stored in a database, including the credential number within the credential, and is provided to the user.

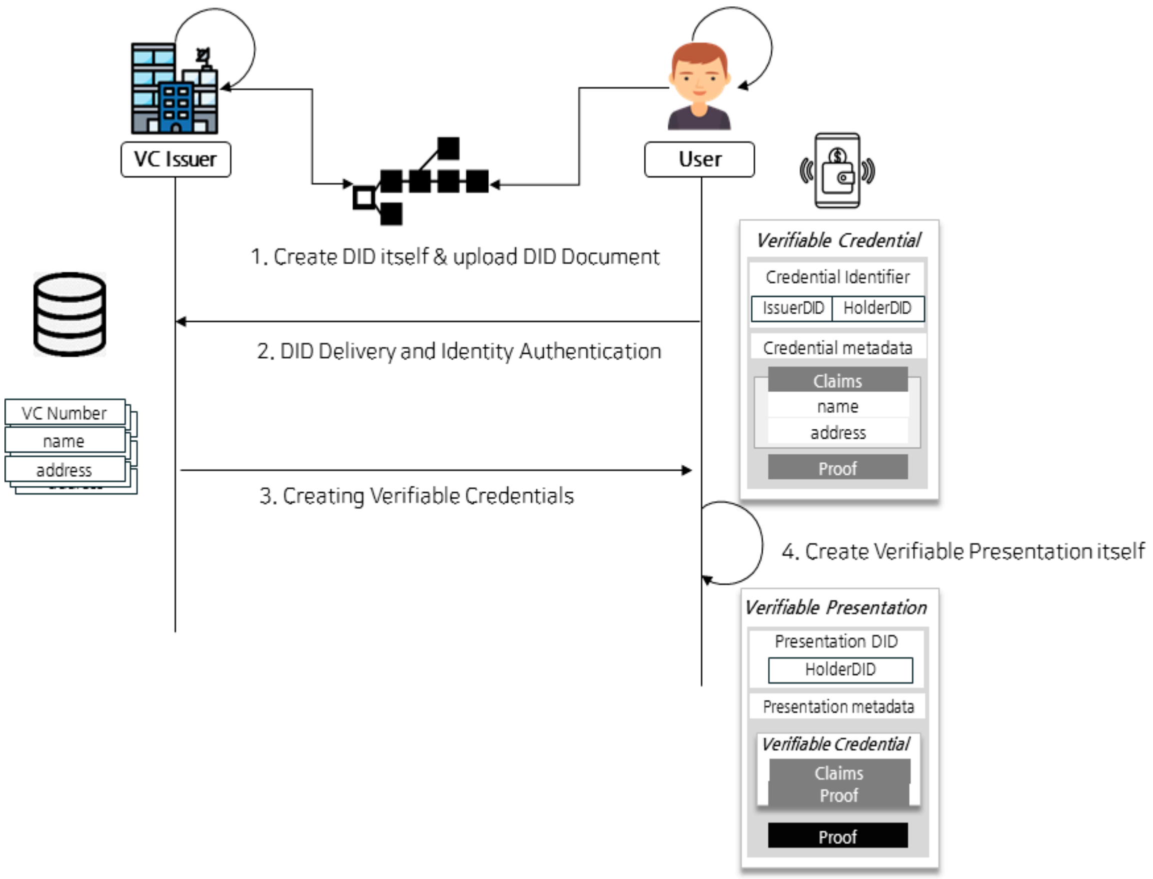


Figure 5. Process of creating credential and presentation through DID.

The credential issued by the issuer should represent the user's identity within the metaverse. Therefore, it can include various items identifying the user, such as a resident registration card, driver's license, diploma, and other relevant documents, as shown in Table 1. Afterward, a holder's signature is included in the Credential, as shown in Table 2, to generate the presentation.

Table 1. Structure of verifiable credential.

Verifiable credential {credential DID, issuer DID, issuance date, (claim 1, claim 2, . . ., claim n), proof(issuer's signature)}

Table 2. Structure of verifiable presentation.

Verifiable presentation {presentation DID, presentation metadata, verifiable credential, proof (holder's signature)}

When mapping users' credentials or personal information to credentials and presentations using DID, there exists a privacy concern where all information, including unnecessary details, is revealed. To address this, it is essential for users to employ the selective disclosure method, allowing only necessary information to be disclosed. In this scheme, the CL signature [31] is utilized to implement this approach. The CL signature can be considered probabilistically secure under the strong RSA assumption. During the signature verification, using the Sigma protocol for the information intended to be concealed, its value can be calculated.

3.4. ZKP Tools

Our tool allows users to generate proof through ZKP using presentations, and automatically generates a smart contract to verify these proofs. This tool utilizes the Pinocchio [32] algorithm to create smart contracts based on ZK-SNARKs for implementing ZKP. It uses the ZoKrates [33] tool for handling ZKP to generate users' proof and smart contracts. The process of generating proof through the corresponding tool, as shown in Figure 6, is as follows.

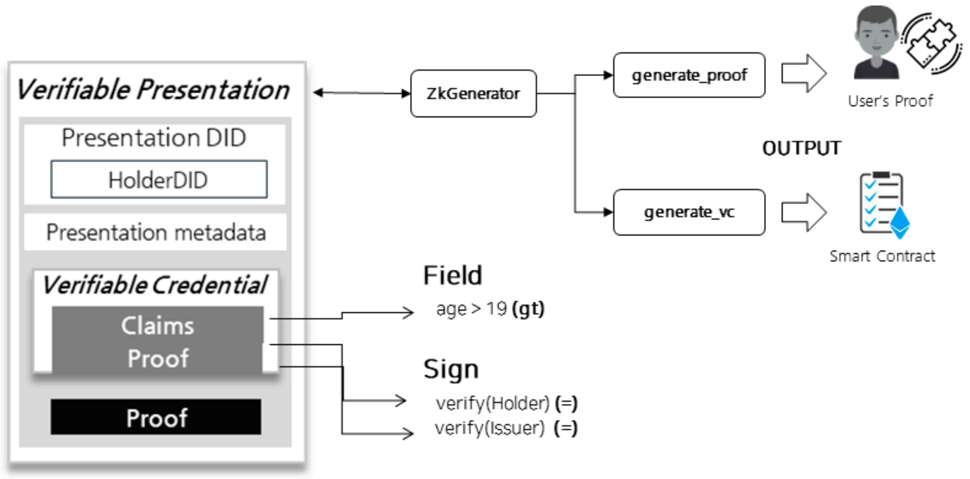
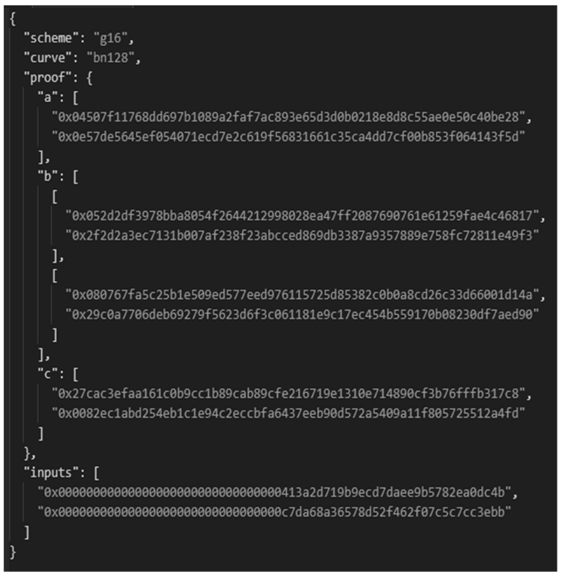


Figure 6. Presentation to proof.

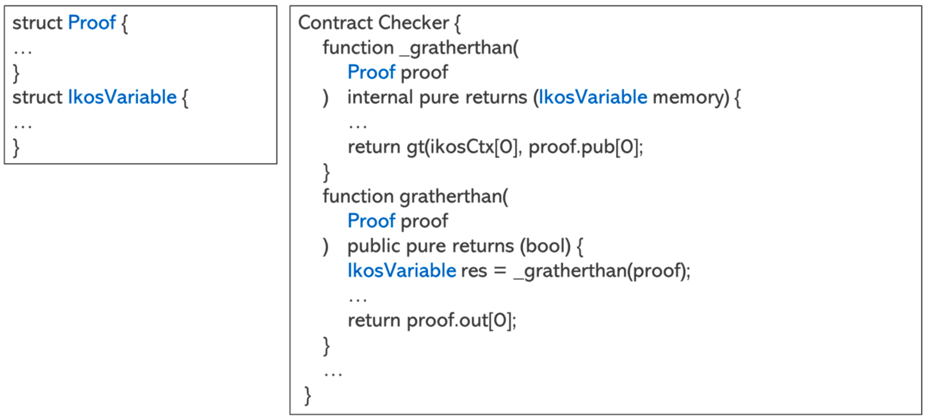
(1) The ZKP tool takes the user's presentation as input, parses it, and separates the field and sign components for further processing.
(2) The validator performs operations such as verifying the DID signature (CL signature), implementing the "greater than" operation for age validation (19 years and above), and executing the "equal" operation.
(3) The ZoKrates library is used to implement proof and verification functions.
(4) The generator provides the user with the proof value of the presentation. The validator uses the verify function provided by the smart contract to verify the true/false value of the user's proof.

The proof generated by the signature verification and age is shown in Figure 7. When a valid proof is provided, it reflects a true value, allowing the validation of the user's value. When calling a function to verify a value in a smart contract, a function, prefixed with "_", is called, and the circuit of the function called when creating a circuit is added. When the verification function is executed, it determines the truth or falsehood of the proof. If the proof is true, the received result from the 'out' value is directly returned, enabling the flow to be similar to invoking a regular function. This allows for a seamless utilization. If the proof is false, the function executes an assert statement to cancel the execution. In Figure 8, the proof generation function pertains to the calculations for "greater than" operations based on the value of the proof.



```
{
  "scheme": "g16",
  "curve": "bn128",
  "proof": {
    "a": [
      "0x04507f11768dd697b1089a2faf7ac893e65d3d0b0218e8d8c55ae0e50c40be28",
      "0x0e57de5645ef054071ecd7e2c619f56831661c35ca4dd7cf00b853f064143f5d"
    ],
    "b": [
      [
        "0x052d2df3978bba8054f2644212998028ea47ff2087690761e61259fae4c46817",
        "0x2f2d2a3ec7131b007af238f23abcced869db3387a9357889e758fc72811e49f3"
      ],
      [
        "0x080767fa5c25b1e509ed577eed976115725d85382c0b0a8cd26c33d66001d14a",
        "0x29c0a7706deb69279f5623d6f3c061181e9c17ec454b559170b08230df7aed90"
      ]
    ],
    "c": [
      "0x27cac3efaa161c0b9cc1b89cab89cfe216719e1310e714890cf3b76fffb317c8",
      "0x0082ec1abd254eb1c1e94c2eccbfa6437eeb90d572a5409a11f805725512a4fd"
    ]
  },
  "inputs": [
    "0x00000000000000000000000000000000413a2d719b9ecd7daee9b5782ea0dc4b",
    "0x00000000000000000000000000000000c7da68a36578d52f462f07c5c7cc3ebb"
  ]
}
```

Figure 7. Signature verification and proof generated through age.



```
struct Proof {
...
}
struct IkosVariable {
...
}
```

```
Contract Checker {
    function _gratherthan(
        Proof proof
    )  internal pure returns (IkosVariable memory) {
        ...
        return gt(ikosCtx[0], proof.pub[0];
    }
    function gratherthan(
        Proof proof
    )  public pure returns (bool) {
        IkosVariable res = _gratherthan(proof);
        ...
        return proof.out[0];
    }
    ...
}
```

Figure 8. Solidity code for generating proofs using the "greater than" function.

Using this tool, users can input credentials (in JSON format) stored in their wallet. Subsequently, users can invoke the verify contract procedure to choose the type of proof they want to generate. Later, when the user requests a specific ZKP (e.g., age over 19), the tool parses this information and generates the corresponding proof. With the generated proof, users can undergo verification by the verifyProof entity below and receive the SBT issuance. Hence, users can enhance their convenience as they can create proofs for desired values using only the credentials within their wallet and the contract, without needing to know the details of any ZKP encryption algorithms.

3.5. SBT Minting Contract

The operational process of the contract is shown in Figure 9. The zok-verify contract verifies the user's presentation proof and then issues the SBT via the contract. Only the designated the SBT issuer can issue an SBT, and it only works if the value of the user's proof is true.

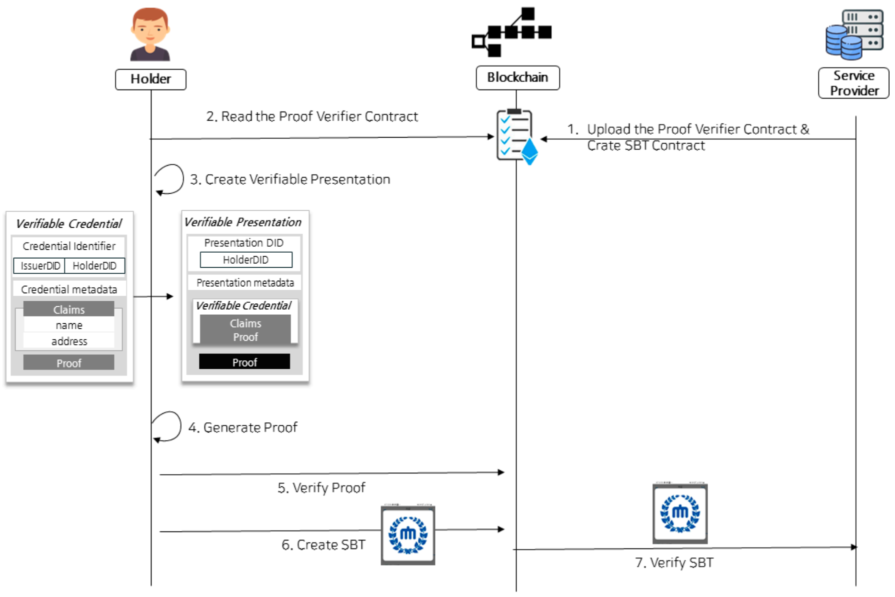
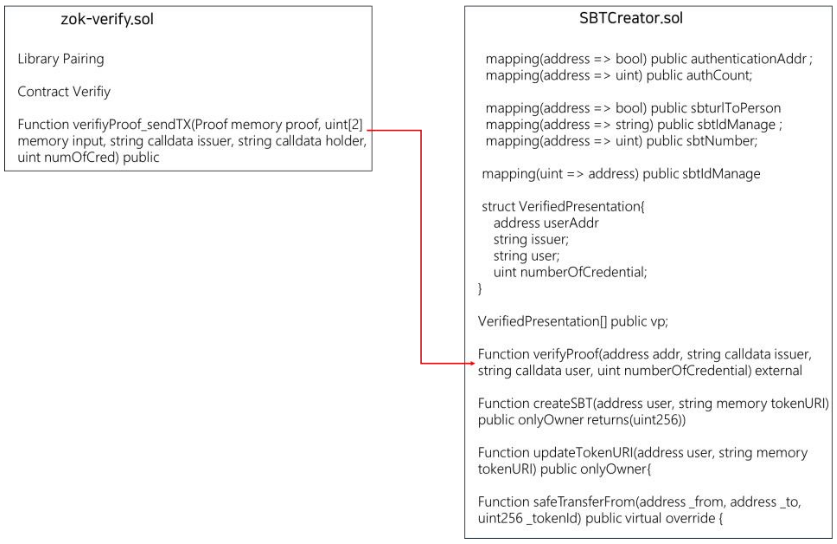


Figure 9. Contract operation process.

In the case of the SBT contract, it uses the ERC721 standard from OpenZeppelin to generate the SBT and includes the "cannot transfer" condition to prevent it from being transferred to anyone else. The appearance of the contract is shown in Figure 10. The contract is composed of two contracts: zok-verify, which verifies the user's proof, and SBTCreator, which issues the SBT. In the zok-verify contract, verifyProof_sendTX() must be called first, and only if the proof is true, can VerifyProof() in the SBTCreator Contract be invoked. SBT issuance is possible only when these requirements are satisfied.



```
zok-verify.sol

Library Pairing

Contract Verifiy

Function verifiyProof_sendTX(Proof memory proof, uint[2]
memory input, string calldata issuer, string calldata holder,
uint numOfCred) public
```

```
SBTCreator.sol

  mapping(address => bool) public authenticationAddr ;
  mapping(address => uint) public authCount;

  mapping(address => bool) public sbturlToPerson
  mapping(address => string) public sbtIdManage ;
  mapping(address => uint) public sbtNumber;

 mapping(uint => address) public sbtIdManage

 struct VerifiedPresentation{
    address userAddr
    string issuer;
    string user;
    uint numberOfCredential;
}

VerifiedPresentation[] public vp;

Function verifyProof(address addr, string calldata issuer,
string calldata user, uint numberOfCredential) external

Function createSBT(address user, string memory tokenURI)
public onlyOwner returns(uint256))

Function updateTokenURI(address user, string memory
tokenURI) public onlyOwner{

Function safeTransferFrom(address _from, address _to,
uint256 _tokenId) public virtual override {
```

Figure 10. The zok-verify and SBTCreator contracts.

In Algorithm 1, the verifyProof_sendTX() function within the Verify Contract procedure verifies the user's presentation proof. This function is invoked by supplying the proof generated in Section 3.4 and the issuer and user's DID present in the presentation and the credential number as inputs. The verify function is generated using the ZoKrates plugin. In this context, when the proof value submitted by the user is true, the verifyProof() function within the SBTContract confirms that the presentation proof has already been successfully validated. To prevent the possibility of other users using the same proof value, the duplicatedProof array is utilized to store proof values.

Algorithm 1. verifyProof_sendTX() [Verify.Contract]

```
Input: calldata proof, calldata input, calldata issuer, calldata user, uint numOfCred
  If (verify(proof, input) == true)
      SBTContract.verifyProof(msg.sender, issuer, holder, numOfCred)
                                  ▷ verify presentation proof using ZoKrates
                                    If presentation proof is true, call the SBTCreatContract
duplicatedProof[numberOfProof] ← proof
                                  ▷ Storing a list for duplicate confirmation.
```

In this contract, ZK-SNARKs, as used in ZoKrates, are employed. Generating proof incurs relatively high costs, but in the verification process, it maintains O(1) complexity. Therefore, even for significant computations within the smart contract, solutions are always resolved in constant time, resulting in a minimal GAS overhead. Using Solidity version ˆ0.8.0, deploying the verifyProof_sendTX() function costs GAS 1,336,311, and calling the verify function consumes GAS 227,578. Currently, it verifies three proofs, and even with an increase in the number of proofs, the GAS cost remains at GAS 227,578.

In the SBTContract, the verifyProof called from the preceding Verify Contract procedure is used to register the user's address in the authenticationAddr. As in Algorithm 2, the verifyProof function can only be invoked within the Verify Contract procedure. For future audits, the verified information of the credential is stored in the vp array for management purposes.

Algorithm 2. verifyProof [SBTContract]

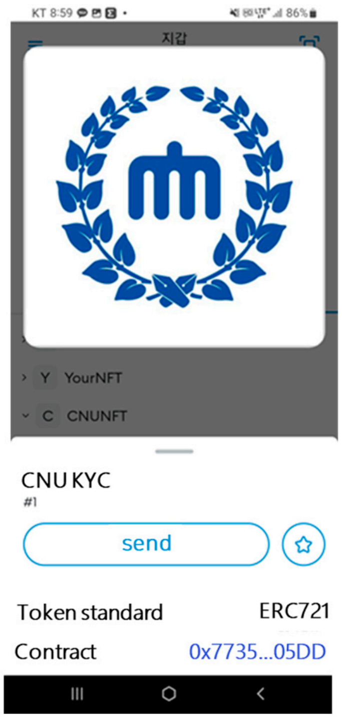
```
Input: string tokenurl, address user, string issuer, string user, uint numOfCred
  onlyVerfy.Contract              ▷ Can only be executed within the Verify Contract.
  authenticationAddr[user] ← True ▷ Indicates that the user has been authenticated
                                    with proof from the Verify Contract.
  Vp ← push(VerifiedPresentation(addr, issuer, user, numOfCred)
                                  ▷ To express gratitude, information is added to the vpList.
  authCount[addr] ← vp.length;    ▷ Manages the sequence of users authenticated.
```

The following is the createSBT() function, responsible for issuing an SBT, as outlined in Algorithm 3. This function can only be accessed by the issuer and can only be issued to the person whose proof, which was previously registered in the authnticationAddr, has been verified. Additionally, only one SBT can be issued per address. The SBT is created by calling the mint function through ERC721.sol. The issuer generates a tokenURI and uses it to create the SBT.

Algorithm 3. createSBT()

```
Input: string tokenurl, address user
  onlyOwner()                     ▷ Only administrators are authorized to issue SBT.
  require(authenticationAddr[user], "Not Authentication person");
                                  ▷ SBT can only be issued to individuals who have
                                    successfully undergone proof verification.
  require(sbtToPerson[user], "already exist");
                                  ▷ Only one SBT can be issued per individual ID.
  tokenIds.increment();
  uint256 newItemId = tokenIds.current();
  _mint(user , newItemId);        ▷ Issues to the user, including the SBT number
  _setTokenURI(newItemId , tokenURI) ;▷ Setting up SBT and Token URI
  sbtToPerson[user] = true;       ▷ Adding SBT issuing address
  sbturlToPerson[user] = tokenURI; ▷ Adding tokenURI for Quick Search
  sbtNumber[user] = newItemId;    ▷ Adding token number
  Output: newItenId;
```

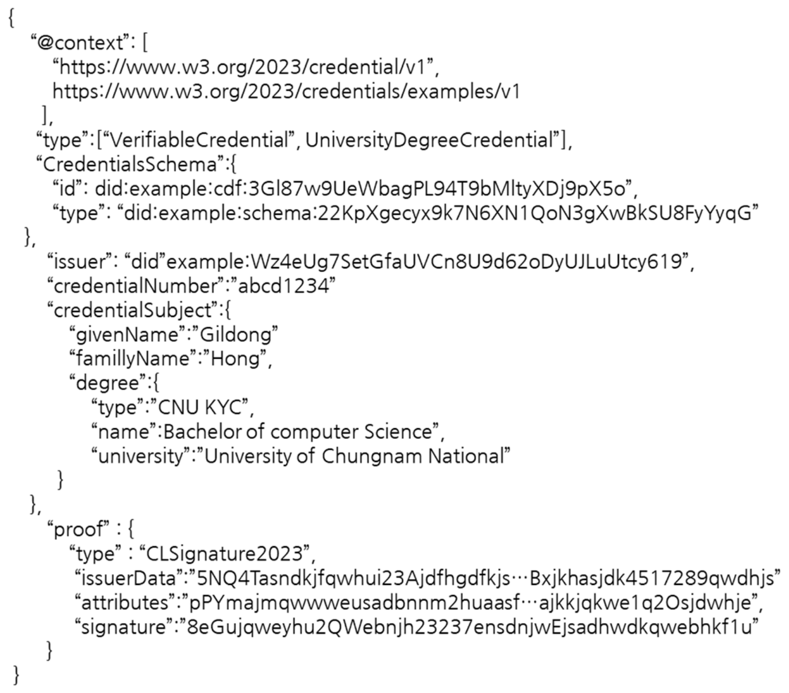
The generated SBT appears as shown in Figure 11a, and its corresponding JSON structure is depicted in Figure 11b. Subsequently, functionality is implemented to facilitate the SBT management. This includes searching by address to determine the SBT issuance status, associated tokenURI information, and the token number.



CNU KYC
#1

Token standard ERC721
Contract 0x7735...05DD

(**a**)



```
{
  "@context": [
    "https://www.w3.org/2023/credential/v1",
    https://www.w3.org/2023/credentials/examples/v1
   ],
  "type":["VerifiableCredential", UniversityDegreeCredential"],
  "CredentialsSchema":{
    "id": did:example:cdf:3Gl87w9UeWbagPL94T9bMltyXDj9pX5o",
    "type": "did:example:schema:22KpXgecyx9k7N6XN1QoN3gXwBkSU8FyYyqG"
 },
    "issuer": "did"example:Wz4eUg7SetGfaUVCn8U9d62oDyUJLuUtcy619",
    "credentialNumber":"abcd1234"
    "credentialSubject":{
      "givenName":"Gildong"
      "famillyName":"Hong",
      "degree":{
        "type":"CNU KYC",
        "name":Bachelor of computer Science",
        "university":"University of Chungnam National"
      }
  },
    "proof" : {
      "type" : "CLSignature2023",
      "issuerData":"5NQ4Tasndkjfqwhui23Ajdfhgdfkjs…Bxjkhasjdk4517289qwdhjs"
      "attributes":"pPYmajmqwwweusadbnnm2huaasf…ajkkjqkwe1q2Osjdwhje",
      "signature":"8eGujqweyhu2QWebnjh23237ensdnjwEjsadhwdkqwebhkf1u"
    }
}
```

(**b**)

Figure 11. (**a**) CNU KYC SBT in Wallet; (**b**) Credential JSON associated with SBT.

While specifications for the non-transferable SBT, like ERC-1138 and ERC-5192, are under development, we have implemented code in a way that makes the transferFrom operation fundamentally impossible, as described in Algorithm 4. As a result, no trading is possible for any SBT generated through the SBTContract.

Algorithm 4. transferFrom(_form, _to, _tokenId)

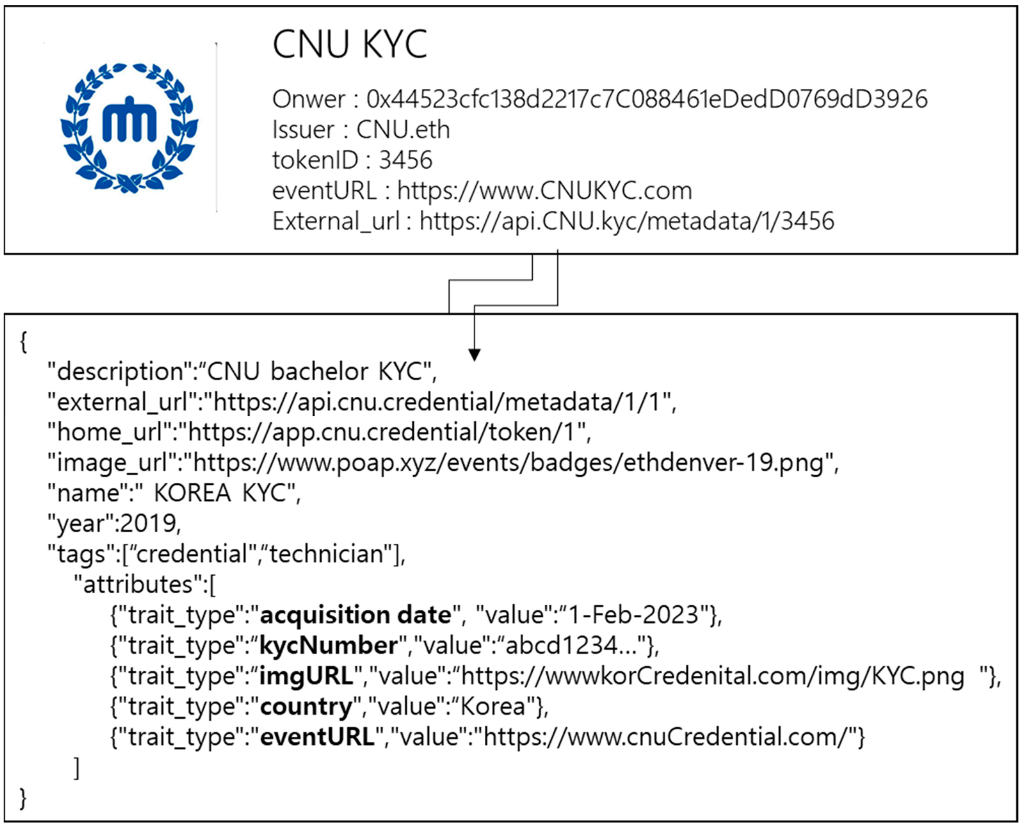
```
Input: address _form, addres _to, uint256 _tokenID
require(cannotTransfer == false, "You cannot transfer this SBT")
                    ▷ Issued SBT cannot be transferred to other individuals.
```

Lastly, in Algorithm 5, there is the updateURL() function. This function can only be called by issuers and is intended to allow the modification of the tokenURI for previously created SBT in response to user requests. This function allows SBT present in user wallet addresses to be granted functionality similar to revocation.

Algorithm 5. updateURL()

```
Input: string tokenurl, string tokenURI
onlyOnwer()                         ▷ Only administrators have the authority to
                                       update SBT
_setTokenURI(sbtNumber[user], tokenURI)  ▷ By altering the tokenURI of an SBT,
                                           the content within it can be modified.
```

The blockchain stores owner, issuer, and TokenURL information, while the metadata in External_url is stored on the issuer's server. User information, excluding personal data, is exposed, addressing privacy concerns. In the future, if an individual who possesses a specific SBT engages in criminal activities, it becomes possible to obtain information from the issuer who issued the corresponding credential. Using the kycNumber, one can access and review the details of the issued credential, as depicted in Figure 12.



CNU KYC

Onwer : 0x44523cfc138d2217c7C088461eDedD0769dD3926
Issuer : CNU.eth
tokenID : 3456
eventURL : https://www.CNUKYC.com
External_url : https://api.CNU.kyc/metadata/1/3456

```
{
   "description":“CNU bachelor KYC",
   "external_url":"https://api.cnu.credential/metadata/1/1",
   "home_url":"https://app.cnu.credential/token/1",
   "image_url":"https://www.poap.xyz/events/badges/ethdenver-19.png",
   "name":" KOREA KYC",
   "year":2019,
   "tags":[“credential",“technician"],
      "attributes":[
          {"trait_type":"acquisition date", "value":“1-Feb-2023"},
          {"trait_type":“kycNumber","value":“abcd1234..."},
          {"trait_type":“imgURL","value":“https://wwwkorCredenital.com/img/KYC.png  "},
          {"trait_type":"country","value":“Korea"},
          {"trait_type":"eventURL","value":"https://www.cnuCredential.com/"}
      ]
}
```

Figure 12. Generated CNU KYC Information.

3.6. DID & Credential Integrated Wallets

A wallet for storing the SBT received after the user verifies the proof is shown in Figure 13. While it is possible to store the SBT using existing cryptocurrency wallets, the presentation and credential used to obtain the SBT may have been securely stored by the user in another wallet. In this paper, a wallet is proposed that not only facilitates the storage of the SBT but also integrates and manages crypto wallets all in one place. For cryptocurrency wallets, the current implementation uses the Metamask API. For the DID wallet, it follows the requirements outlined in the W3C's upcoming Universal Wallet 2020 standard. The wallets use the HD (Hierarchical Deterministic) wallet structure, which uses the BIP-44 [34].

The key derivation algorithm allows for the derivation of various sub-level private keys from a single seed value. In this process, the secp256k1 curve [35] of the ECDSA algorithm is utilized to generate a public key corresponding to the private key. Through hashing, the address of the crypto wallet is then generated. Similarly, in the case of DID, the private key is derived from the seed value, and the Ed25519 curve [36] of the EdDSA algorithm is used for generating the corresponding public key. This process leads to the creation of a DID-specific identifier, which is then used to generate the DID itself. By using a single wallet that holds the seed value, both the metaverse wallet and the DID wallet are integrated and managed. The HD wallet structure follows the BIP-44 algorithm for these operations. Users have the capability to select a blockchain network within Metamask. In the metaverse wallet address, various items, such as ERC-721 NFTs, ERC-20 tokens, assets, and SBTs can be stored. In the DID wallet, users can not only access the DID document, but also verify the credentials issued with that particular DID. Additionally, the user wallet offers the functionality to generate presentations. By selecting the desired claims, users can create presentations containing their signatures. These presentations can be used to generate proof and include functions to verify this proof.

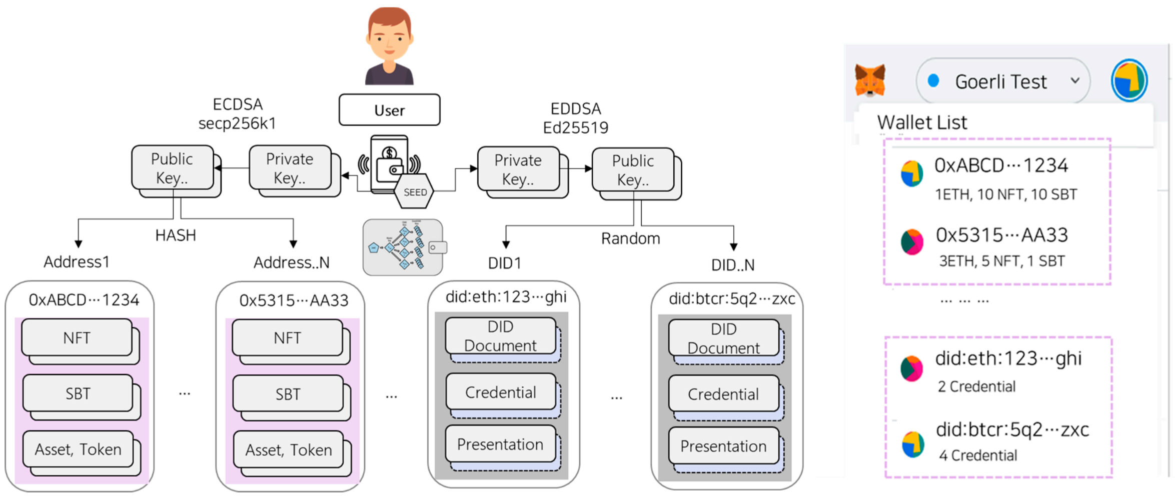


Figure 13. The appearance of an integrated wallet.

In fact, utilizing the EdDSA algorithm, which involves a relatively simpler and more efficient signature generation and verification process compared to the ECDSA algorithm, for wallet address and DID generation, would be a favorable approach. However, presently, Bitcoin wallets generate addresses and transaction signatures based on the ECDSA algorithm. To support various multi-blockchains, both ECDSA and EdDSA algorithms are used. Therefore, by generating identical seeds and creating ECDSA and EdDSA private key pairs based on each seed value, it is possible to support multi-addresses using a hierarchical structure. This approach offers the advantage of deriving multiple addresses from a single seed. Additionally, by leveraging the open-source code of Metamask, adding additional networks is inherently possible. The network includes Ethereum as the default base, and Ethereum test networks such as Goerli are inherently registered. DID, being based on W3C standards, poses no compatibility issues. Regarding address wallets, the SBT that exists on networks such as Ethereum, Polygon, and Optimism, is forked from Ethereum. Hence, additional development is required when dealing with SBTs generated using other platforms, which could be considered a drawback.

3.7. Managing Metaverse Avatar Permissions through the Wallet

Once SBTs are generated, they can be linked to the metaverse avatar and wallet address. To access specific services, the presence of the corresponding SBT is verified within the avatar linked to the user's wallet. This verification process determines whether the user can use services within the metaverse. For such services to be realized, it is necessary to develop a digital wallet address system, as shown in Figure 14. Currently, digital wallets can only store tokens such as NFTs, badges, and assets, and lack support for DIDs, credentials, and presentations. To solve this, a wallet program must be developed that allows users to link their DIDs to their digital wallets, enabling the creation of services where NFTs can be issued using credentials from issuers.

Currently, integration with the metaverse system operating on Ethereum-based networks is feasible. The smart contracts generated to verify proof can only be validated within EVM (Ethereum Virtual Machine)-based networks. Consequently, these proofs can be verified exclusively within networks operating primarily on the Ethereum platform.

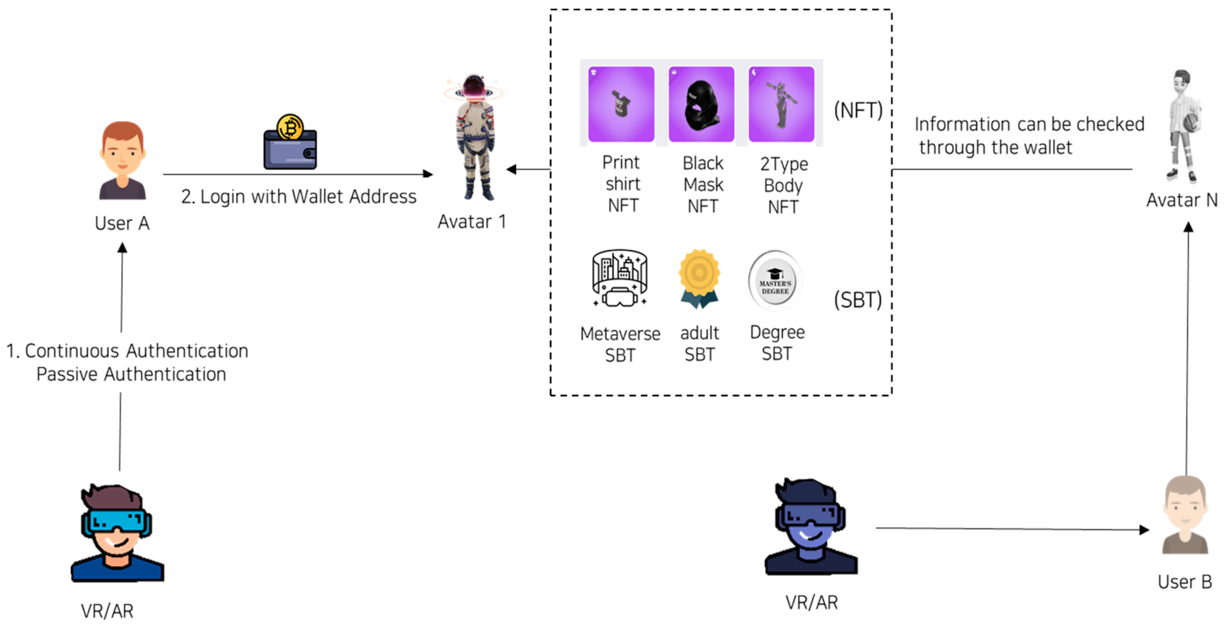

User A
2. Login with Wallet Address
Avatar 1
Print shirt NFT
Black Mask NFT
2Type Body NFT
(NFT)
Metaverse SBT
adult SBT
Degree SBT
(SBT)
Information can be checked through the wallet
Avatar N
1. Continuous Authentication Passive Authentication
VR/AR
VR/AR
User B

Figure 14. Managing metaverse avatar permissions through the wallet.

4. Privacy and Security Analysis

(1) Privacy Analysis

In existing systems, such as metaverses or DeFi, service providers hold all user information, which leads to significant risks in cases of hacking. However, in the proposed system, only the DID issuer (who may also perform auditing later) has user information. Service providers only verify evidence derived from presentations. This setup ensures that users' privacy is protected, as the service providers do not have access to any personal data but can check whether someone meets certain criteria based on the presented proof. If someone who has received an SBT from a service provider engages in money laundering or illegal activities, their suspicious actions can be traced by requesting information about the corresponding credential from the DID issuer using the credential number stored in the SBT external link. In addition, the system includes SBT revocation functionality. Given the potential risk of inferring the identity of a particular individual from different SBTs present in a user's wallet address, the system uses ERC-4906 and ERC-5185 in response to user requests. This allows the metadata of a user's SBT to be updated, effectively breaking the link, and protecting privacy.

(2) Security Analysis

In the proposed protocol, since credentials are not directly mapped to the user's wallet, there exists a potential issue where proofs could be replicated and used. The generated proof only contains a signature asserting that the owner of the corresponding DID is authentic. Thus, if a user provides the same proof as the one used to generate a presentation proof, it might be assumed that they possess the associated credential. To address this, in the current protocol, when a service provider issues an SBT, the proof is stored in a smart contract. If the same proof is submitted again, the protocol prevents the issuance of the SBT. The developed smart contract follows the Ethereum standard ERC-721 and uses the OpenZeppelin library to implement non-transferable functionality, ensuring that proof cannot be transferred. The verification of proof is achieved using ZK-SNARKs from ZoKrates. Both OpenZeppelin and ZoKrates have undergone security validation, ensuring their safe utilization within the smart contract.

(3) ZKP tools Analysis

Using the Rust language, one can use range proofs or arithmetic circuits to verify the contents within a user-generated presentation, such as age and signature. The ZoKrates tool provided by Remix facilitates this process. The setup phase of the circuit is conducted off-chain, with the results shared in the verification contract. When a user wants to prove a statement using their secret value, they use the ZoKrates language and the proving key generated in the setup, along with the secret value, to create a proof. Verification of the proof is possible on the blockchain by deploying the verification contract created with the verification key from the setup, allowing validation of the proof's correctness on the blockchain. By using this tool, the ZoKrates tool provided by Remix can be transformed into a language that generates proof for users, while the smart contract provides the verification contract. in this way, even developers without in-depth knowledge of ZKP can verify the identity of users, making the process accessible and advantageous.

5. Related Works

5.1. DID/Credential Verification Procedures Using Blockchain

Sovrin [37] is an SSI-based system built on the Hyperledger framework, implementing a decentralized PKI using DIDs. On the other hand, uPort [38] operates on the Ethereum platform, utilizing uPort identities for Ethereum account addresses. However, both Sovrin and uPort validate DIDs on the blockchain, lacking sufficient on-chain privacy protection measures. CredChain [39] has introduced a self-sovereign identity framework for secure issuance, sharing, and the verification of credentials, proposing mechanisms to ensure the trustworthiness of credential verification systems and privacy control. CredenceLedger [40] employs permissioned blockchains and cryptographic technologies tailored for the education sector, making verification easy and emphasizing user-centricity and privacy protection through innovative protocols [41]. While these specific domain-oriented Dapps propose privacy protection solutions, they come with the drawback of needing to validate DIDs for each required service. This paper [42] presents strategies to address centralization issues, and in this paper [43], integrating DID, off-chain verifiable credentials, and on-chain smart contracts is proposed. However, the on-chain signature verification method for DIDs is not clearly outlined, and approaches for managing user credentials are not discussed. Additionally, Cerberus [44] performs credential revocation and verification processes using multiple addresses, considering the revocation process as an intermediate step in the SBT recovery process. The Coconut [45] solution, as a smart contract library for Chainspace and Ethereum, ensures confidentiality and integrity even if the issuing authority is malicious and allows specific attributes to be revealed to verifiers. However, these papers, while verifying DIDs on-chain, have an advantage, but require submitting DIDs for each service. Moreover, in the context of using Web3, they do not explain the relationship between DID and the user's wallet, as they authenticate users using the wallet, not DID, which can be considered a disadvantage.

5.2. Definition of Identity in Blockchain

According to the definition of identity [46] in the blockchain, identity is categorized into transaction identity, based on wallet account information, persona identity, rooted in personal characteristics, credential identity, based on social behavior or asset ownership, reputation identity, established from reputation, and data identity, built upon actual data.

Credential systems, such as POAP [47] and Ethereum Name Service [48], authenticate identity based on social behavior and asset ownership. These systems typically involve issuing NFTs to wallet accounts, similar to SBTs, for identity verification. However, existing NFTs can be transferred, making it challenging to solely rely on NFT ownership for KYC verification. Furthermore, NFTs often cannot hold all of the user's information, and the drawback lies in the fact that issuers of NFTs need to store user information in databases operated by each issuer for user authentication.

Furthermore, individual identification systems like BrightID [49], Proof of Humanity [50], and WorldCoin [51] (Proof of Personhood) establish that the owner of a wallet is indeed a real person. BrightID leverages linkage to social accounts to confirm identity, while WorldID in WorldCoin uses iris scans linked to wallet accounts to verify personhood. These approaches prevent Sybil attacks and aim to function as UBI (Universal Basic Income) tokens. However, BrightID relies on a user's social network rather than their exact identity, which falls short of fully satisfying KYC requirements. WorldID raises concerns regarding security due to the usage of users' iris information.

Project Galaxy [52] ID relies on off-chain reputation data, such as credit scores, for identity verification. Rabbit Hole [53] verifies users based on their Web3 activity history, showcasing their knowledge and experience through data. Reputation data is not suitable as an identity proof since it does not represent KYC processes but rather only certain score indicators. Finally, there is data-based authentication, like DID. However, this approach faces privacy issues due to the transparent nature of blockchains for identity verification. Moreover, to support various blockchains, certifiers would need to issue verifiable credentials for each blockchain account, leading to privacy concerns, as certifiers would know the asset balances and transaction history of each user's wallet account.

5.3. Final Review Analysis

In Section 5.1, we analyzed the use of DID in the context of the blockchain. In most cases, the blockchain acts as a ledger for DID, and the advantage lies in on-chain verification rather than off-chain. However, the drawback is that users have to submit DID every time verification occurs for different services. Additionally, when utilizing Web3, the authentication is done using the user's wallet, not DID. This creates a challenge in explaining the relationship between DID and the user's wallet. In Section 5.2, we classified identities within the blockchain into a transactional identity based on the wallet account information, character identity based on traits, qualification identity based on social behavior or asset ownership, reputation identity based on reputation, and data identity based on actual data. We compared these identities, highlighting their respective strengths and weaknesses in Table 3. As mentioned in our paper, the proposed scheme is suitable for connecting with metaverse avatars.

Table 3. Comparison of identity in blockchain.

	Credential Identity System	**Personality Identity System**	**Reputation Identity System**	**Decentralized Identity System**	**Proposed Scheme**
Related work	[47,48]	[49–51]	[52]	[37–45]	This paper
Is verification conducted on-chain?	Yes	No	Yes	No	Yes
Is the identity token provided?	No	Yes	Yes	No	Yes
Is the wallet account anonymous to the verifier?	No	No	No	Yes (Selective disclosure)	Yes (Selective disclosure)
Is off-chain identity compatible with the wallet?	Yes	Yes	No	No	Yes

6. Discussion

In the DID/credential verification procedures using blockchain parts, the blockchain is simply used as a verification center to use DID, and DID is not verified by smart contracts or used as a KYC procedure, so a direct comparison with this paper's proposed scheme is not possible. And in the case of identity in the blockchain, it can be said that it is far from the KYC process, as it only suggests a method of expressing identity in the blockchain using various methods. Accordingly, we would like to compare papers on measures to ensure KYC processes in a blockchain environment.

Firstly, the author of [54] discusses SBT in the future Web3 ecosystem, making distinctions from credentials. They predict the use of an SBT instead of a credential for verifying user identities in Web3, and highlight the challenges faced in the non-fungibility of SBTs. However, this paper does not mention KYC processes through SBTs, and, given its futuristic outlook on the world of SBTs, it might not directly compare well with our paper. On the other hand, this paper [55] introduces the KYCE (Know Your Customer on Ethereum) system, which is a system where identity providers store the credentials of users who have successfully completed KYC verification in an encrypted accumulator. Witnesses are then provided to the users. However, including these witnesses in each transaction datum incurs additional costs, and the process of authenticating users before KYC verification is not precisely outlined. Furthermore, it manages user authentication off-chain and only later manages it through smart contracts in a whitelist manner. Additionally, the KYC Privacy-Preserving Identity Scheme is discussed by [46]; KYC processes handled by VASPs using non-fungible tokens for identity verification are discussed. These tokens are associated with approved institutions and utilize BBS+ signatures for ZKP. However, in this approach, VASPs need to calculate values in the Merkle tree using smart contracts and individually relay the calculated values to each VASP to use ZKP, limiting its application. To address these limitations, the same research team submitted a more advanced paper [56]. This solution sets up third-party verifiers to authenticate user identity witnesses and provides interfaces to access whitelists of verified wallet accounts. This eliminates the need for numerous small VASPs to repeatedly verify user identity witnesses. However, in this method, identities are managed by whitelists within smart contracts, requiring the presence of a third party to oversee the whitelist. This method does not utilize DID for user authentication, conducts verification on-chain, which incurs significant costs, and only proposes a process for simple SBT minting.

In contrast, our proposed system utilizes existing DIDs/credentials for verifying user identities. Rather than using off-chain validation like in previous research proposals, our system utilizes ZKP to protect privacy, and verifies users on-chain, ensuring transparency in the verification process. Additionally, we provide a tool for creating ZKP and generating verification contracts, making it user-friendly even for those unfamiliar with ZKP credentials; in addition, generated SBT storage is facilitated through an integrated wallet.

However, our tool faces potential challenges: firstly, it is not a fully decentralized system, as it relies on trusted issuers for creating DID credentials. Secondly, the smart contract currently only operates within the Ethereum-based EVM, limiting its use to platforms supporting smart contracts and SBTs, like Ethereum, while Web3 platforms, such as metaverse systems, operate in diverse blockchain environments. Lastly, there is a compatibility issue between wallet addresses and DID credentials. Currently, the system issues an SBT to wallet addresses based on credentials for user authentication, and the credentials used are stored in contracts in a whitelist form to prevent duplicate use. If the same credentials are used in different contracts, issues might arise, indicating the need for a fundamental solution for compatibility.

The scheme was compared with KYC schemes according to several papers, as shown in Table 4.

Table 4. Comparison of KYC schemes.

	Identity Authentication Factor	KYC Authentication	Smart Contract Utilization	Token Usage	Tool Support	Wallet Support
Privacy-preserving KYC on Ethereum [55]	ID card, passport…	On-chain	Whitelist	X	X	X
A Privacy-Preserving KYC-Compliant Identity Scheme [46]	ID card, passport…	On-chain	Credential Verification (VASP calculation is required)	SBT (Non-Transferable)	X	X
A Universal Privacy-Preserving Multi-Blockchain Aggregated [56]	ID card, passport…	On-chain	Credential Verification (+Third Party)	SBT (Non-Transferable)	O	X
Tbdex [7]	DID/credential	Off-chain	Whitelist	X	X	X
Proposed Scheme	DID/credential	On-chain	Credential Verification (ZKP verification)	SBT (Non-Transferable)	O	O

7. Conclusions

Web3 is a decentralized internet technology that relies on a blockchain-based technology stack, allowing for P2P (peer-to-peer) transactions without dependence on centralized platforms or intermediaries. It is expected to become mainstream within the next 10 years, and by 2024, it is projected that 25% of global businesses will integrate existing applications and services into decentralized Web3 applications. Therefore, defining user identities in the context of Web3 is crucial. Given that Web3 is based on a blockchain technology stack, with wallets forming a core part of the ecosystem, attributing identities to wallets holds significant importance. Web3's reliance on blockchain-based wallets for identity authentication underscores the meaningfulness of assigning identities to these wallets.

Web3 VASPs lack proper identity verification processes for users utilizing their services, enabling criminals to anonymously misuse these services for activities like money laundering and illegal forex transactions. As a result of deficient identity management, cryptocurrency-based crimes are rapidly increasing, leading to severe socioeconomic consequences. Requiring VASPs to prove to users that their wallet account identities have been verified can reduce crimes such as fraud and protect investors' rights. By demanding users to prove to VASPs that they have verified identities associated with wallet accounts, criminals would need to use verified accounts to receive funds, thus decreasing crimes like money laundering.

This system proposes a new schema that combines credentials and SBTs to ensure KYC processes. We provide solutions for on-chain DID verification and ensure privacy through ZKP. Moreover, our approach allows VASPs to prove user identities without storing user information, relieving the burden on VASPs. Our solution can lower the barriers for users to utilize Web3. Additionally, it contributes not only to crime prevention but also to crime eradication. For instance, regulatory authorities can trace the identities of suspicious wallet accounts. As mentioned earlier, when users with issued SBTs engage in illegal activities, information about the user can be requested from the issuer who issued the DID. By ensuring unique IDs within the blockchain, this system brings positive outcomes to DAO governance, prevents Sybil attacks, and facilitates airdrops. In the future, we plan to propose solutions for the fundamental issues of this system and develop Dapps applicable to Web3.

Author Contributions: Writing—original draft, G.K.; Supervision, J.R. All authors have read and agreed to the published version of the manuscript.

Funding: (1) This work was supported by Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Government of Korea (MSIT) (2020-0-00321, Privacy Preserving Self-controlled Decentralized Identity Management and Security Technology in 5G Service Environments). (2) This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2023R1A2C1006936).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. The Chainalysis 2022 Crypto Crime Report. Available online: https://hkibfa.io/wp-content/uploads/2023/02/Crypto_Crime_Report_2023.pdf (accessed on 6 August 2023).
2. Hu, M.; Lee, A.D.; Putniņš, T.J. Evading Capital Controls via Cryptocurrencies: Evidence from the Blockchain. 2021. Available online: https://ssrn.com/abstract=3956933 (accessed on 6 August 2023).
3. Roblox Game Hacked, 100 Million Users' Data Compromised. Available online: https://www.expresscomputer.in/security/roblox-game-hacked-100-million-users-data-compromised-report/55078/ (accessed on 6 August 2023).
4. Decentraland. Available online: https://decentraland.org/ (accessed on 6 August 2023).
5. SandBox. Available online: https://www.sandbox.game/en/about/sand/ (accessed on 6 August 2023).
6. Pilkington, M. Blockchain technology: Principles and applications. In *Research Handbook on Digital Transformations*; Edward Elgar Publishing: Cheltenham, UK, 2016.
7. tbDEX: A Liquidity Protocol v0.2. Available online: https://tbdex.io/whitepaper.pdf (accessed on 6 August 2023).
8. Decentralized Identity Foundation. 2018. Available online: https://identity.foundation/ (accessed on 6 August 2023).
9. Weyl, E.G.; Ohlhaver, P.; Buterin, V. Decentralized Society: Finding Web3's Soul. 2022. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4105763 (accessed on 6 August 2023).
10. Binance to Launch Binance Account Bound (BAB) Token, the First-Ever Soulbound Token on BNB Chain. Available online: https://www.binance.com/en/support/announcement/0fe1e7c8781844e29f56cb674231dfd7 (accessed on 6 August 2023).
11. Giannopoulou, A. Digital Identity Infrastructures: A Critical Approach of Self-Sovereign Identity. *Digit. Soc.* **2023**, *2*, 18.s. [CrossRef]
12. Azbeg, K.; Ouchetto, O.; Andaloussi, S.J. BlockMedCare: A healthcare system based on IoT Blockchain and IPFS for data management security. *Egypt. Inform. J.* **2022**, *23*, 329–343. [CrossRef]
13. Johari, R.; Kumar, V.; Gupta, K.; Vidyarthi, D.P. BLOSOM: BLOckchain technology for Security of Medical records. *ICT Express* **2022**, *8*, 56–60. [CrossRef]
14. EIP-721. Available online: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-721.md (accessed on 6 August 2023).
15. Wood, G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**, *151*, 1–32.
16. Szabo, N. Formalizing and securing relationships on public networks. *First Monday*, 1997.
17. EIP-5192. Available online: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-5192md (accessed on 6 August 2023).
18. EIP-5727. Available online: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-5727.md (accessed on 6 August 2023).
19. EIP-6454. Available online: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-6454.md (accessed on 6 August 2023).
20. Lunesu, M.I.; Tonelli, R.; Pinna, A.; Sansoni, S. Soulbound Token for Covid-19 Vaccination Certification. In Proceedings of the 2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), Atlanta, GA, USA, 13–17 March 2023; pp. 243–248.
21. Nikolić, S.; Matić, S.; Čapko, D.; Vukmirović, S.; Nedić, N. Development of a blockchain-based application for digital certificates in education. In Proceedings of the 2022 30th Telecommunications Forum (TELFOR), Belgrade, Serbia, 15–16 November 2022; pp. 1–4.
22. Goldston, J.; Chaffer, T.J.; Osowska, J.; Goins, C.V., II. Digital Inheritance in Web3: A Case Study of Soulbound Tokens and the Social Recovery Pallet within the Polkadot and Kusama Ecosystems. *arXiv* **2023**, arXiv:2301.11074.
23. Rogaway, P. (Ed.) Advances in Cryptology—CRYPTO 2011. In Proceedings of the 31st Annual Cryptology Conference, Santa Barbara, CA, USA, 14–18 August 2011; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.
24. Yang, X.; Li, W. A zero-knowledge-proof-based digital identity management scheme in blockchain. *Comput. Secur.* **2020**, *99*, 102050. [CrossRef]
25. Barreto, P.L.; Zanon, G.H. Blind Signatures from Zero-Knowledge Arguments. Cryptology ePrint Archive 2023. Available online: https://eprint.iacr.org/2023/067 (accessed on 9 July 2023).

26. Kamel, M.B.M.; Yan, Y.; Ligeti, P.; Reich, C. Attribute Verifier for Internet of Things. In Proceedings of the 2022 32nd International Telecommunication Networks and Applications Conference (ITNAC), Wellington, New Zealand, 30 November–2 December 2022; pp. 1–3.
27. Cao, L.; Wan, Z. Anonymous scheme for blockchain atomic swap based on zero-knowledge proof. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 27–29 June 2020; pp. 371–374.
28. Panja, S.; Roy, B.K. A Secure End-to-End Verifiable e-Voting System Using Zero Knowledge Based Blockchain. Cryptology ePrint Archive 2018. Available online: https://eprint.iacr.org/2018/466 (accessed on 9 July 2023).
29. Murtaza, M.H.; Alizai, Z.A.; Iqbal, Z. Blockchain based anonymous voting system using zkSNARKs. In Proceedings of the 2019 International Conference on Applied and Engineering Mathematics (ICAEM), Taxila, Pakistan, 27–29 August 2019; pp. 209–214.
30. Sahai, S.; Singh, N.; Dayama, P. Enabling privacy and traceability in supply chains using blockchain and zero knowledge proofs. In Proceedings of the 2020 IEEE International Conference on Blockchain (Blockchain), Rhodes, Greece, 2–6 November 2020; pp. 134–143.
31. Guo, N.; Gao, T.; Wang, J. Privacy-preserving and efficient attributes proof based on selective aggregate CL-signature scheme. *Int. J. Comput. Math.* **2016**, *93*, 273–288. [CrossRef]
32. Parno, B.; Howell, J.; Gentry, C.; Raykova, M. Pinocchio: Nearly practical verifiable computation. *Commun. ACM* **2016**, *59*, 103–112. [CrossRef]
33. Eberhardt, J.; Tai, S. Zokrates-scalable privacy-preserving off-chain computations. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1084–1091.
34. BIP-44. Available online: https://github.com/bitcoin/bips/blob/master/bip-0044.mediawiki (accessed on 6 August 2023).
35. Johnson, D.; Menezes, A.; Vanstone, S. The elliptic curve digital signature algorithm (ECDSA). *Int. J. Inf. Secur.* **2001**, *1*, 36–63. [CrossRef]
36. Bernstein, D.J.; Duif, N.; Lange, T.; Schwabe, P.; Yang, B.Y. High-speed high-security signatures. *J. Cryptogr. Eng.* **2011**, *2*, 77–89. [CrossRef]
37. Khovratovich, D.; Law, J. Sovrin: Digital identities in the blockchain era. *Github Commit Jasonalaw Oct.* **2017**, *17*, 38–99.
38. Lundkvist, C.; Heck, R.; Torstensson, J.; Mitton, Z.; Sena, M. Uport: A Platform for Self-Sovereign Identity. 2017. Available online: https://whitepaper.uport.me/uPort_whitepaper_DRAFT20170221.pdf (accessed on 21 October 2023).
39. Mukta, R.; Martens, J.; Paik, H.Y.; Lu, Q.; Kanhere, S.S. Blockchain-based verifiable credential sharing with selective disclosure. In Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 29 December 2020–1 January 2021; pp. 959–966.
40. Arenas, R.; Fernandez, P. CredenceLedger: A permissioned blockchain for verifiable academic credentials. In Proceedings of the 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Stuttgart, Germany, 17–20 June 2018; pp. 1–6.
41. Singh, K.; Dib, O.; Huyart, C.; Toumi, K. A novel credential protocol for protecting personal attributes in blockchain. *Comput. Electr. Eng.* **2020**, *83*, 106586. [CrossRef]
42. Ramachandran, M.; Chowdhury, N.; Third, A.; Domingue, J.; Quick, K.; Bachler, M. Towards complete decentralised verification of data with confidentiality: Different ways to connect solid pods and blockchain. In *Proceedings of the Companion Proceedings of the Web Conference 2020, Taipei, Taiwan, 20–24 April 2020*; pp. 645–649.
43. Casonato, M. Owning Your Data through Self-Sovereign Identity: Agents Implementation for Verifiable Credentials Interaction. 2021. Available online: https://thesis.unipd.it/handle/20.500.12608/34924 (accessed on 21 October 2023).
44. Tariq, A.; Haq, H.B.; Ali, S.T. Cerberus: A blockchain-based accreditation and degree verification system. *IEEE Trans. Comput. Soc. Syst.* **2022**, *10*, 1503–1514. [CrossRef]
45. Sonnino, A.; Al-Bassam, M.; Bano, S.; Meiklejohn, S.; Danezis, G. Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. *arXiv* **2018**, arXiv:1802.07344.
46. Sun, N.; Zhang, Y.; Liu, Y. A privacy-preserving kyc-compliant identity scheme for accounts on all public blockchains. *Sustainability* **2022**, *14*, 14584. [CrossRef]
47. POAP. Available online: https://poap.xyz/ (accessed on 6 August 2023).
48. Ethereum Name Service. Available online: https://ens.domains/ (accessed on 6 August 2023).
49. BrightID. Available online: https://www.brightid.org/ (accessed on 6 August 2023).
50. Proof of Humanity. Available online: https://www.proofofhumanity.id/ (accessed on 6 August 2023).
51. Worldcoin. Available online: https://whitepaper.worldcoin.org/ (accessed on 6 August 2023).
52. Project Galaxy. Available online: https://docs.galaxy.eco/ (accessed on 6 August 2023).
53. Rabiit Hole. Available online: https://rabbithole.mirror.xyz/ (accessed on 6 August 2023).
54. Hildebrandt, F. The future of soulbound tokens and their blockchain accounts. In *Konferenzband zum Scientific Track der Blockchain Autumn School (2022)*; Hochschule: Mittweida, Germany, 2022; pp. 18–24.

55. Biryukov, A.; Khovratovich, D.; Tikhomirov, S. Privacy-preserving KYC on Ethereum. 2018. Available online: https://orbilu.uni.lu/handle/10993/35915 (accessed on 21 October 2023).
56. Sun, N.; Zhang, Y.; Liu, Y. A Universal Privacy-Preserving Multi-Blockchain Aggregated Identity Scheme. *Appl. Sci.* **2023**, *13*, 3806. [CrossRef]