IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Blockchain-Based Autonomous Notarization System Using National eID Card

SHINYA HAGA[1], KAZUMASA OMOTE[1]
[1]Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba 305-8577, Japan
Corresponding author: Shinya Haga (e-mail: s2120545@s.tsukuba.ac.jp).

ABSTRACT In recent years, the spread of information and communication technology has led to the emergence of e-government, which is the electronic replacement of government services. E-government is said to be compatible with blockchain technology, which has led to various studies on the possibility. Notarization, one of the functions of the government, has particularly been examined for the potential adoption of blockchain technology. However, because the notary public must authenticate the document's contents during the notarization process, they have been difficult to replace with smart contracts. In this study, we focus only on fixed date notarizations and propose a fully automated notarization system by combining a national eID card with Public Key Infrastructure and smart contracts. A fixed date is a notarization that allows a notary public to guarantee that a document existed, regardless of the authenticity of the document's content. Therefore, it can be replaced by a smart contract. Specifically, our proposed system automatically authenticates the creator and the document for electronic documents signed with a national eID card and uses the transaction receipt generated when the information is stored on the blockchain as a certificate of notarization. Verification of the signed data is done inside the blockchain by smart contracts, which eliminates the need for a verification authority. We further demonstrate the effectiveness of the proposed method in a Japanese use case as proof of concept.

INDEX TERMS Blockchain, Smart contract, Ethereum, E-government, National eID Card, Notarization System, Authorization

I. INTRODUCTION

In recent years, our lives have improved greatly with the spread of information and communication technologies (ICT). ICT has also been a tremendous boon to government services, which has led us to the new concept of e-government. Supriyanto et al. [1] showed various studies have defined e-government as an ICT application strategy to improve public services with the aim of increasing government interaction with citizens, employees, or any internal entity. A key e-government service is electronic identification (eID). This involves generating an ID on a smart card that allows for digital storage of carrier data, including recognition functions, as well as more complex security measures that allow access to online services for citizens through encryption of personal information [2], [3].

There is also a government process called notarization. Notarization is an official anti-fraud process that assures parties to a transaction that a document is authentic and trustworthy. Lately, an increasing number of studies have tried incorporating blockchain technology into this system. However, most of them have only partially replaced the work of the notary as well as reduced errors and fraud [4]–[6]. Few have tried to fully automate the work of notaries and completely replace them with smart contracts on the blockchain. The few studies [7] that have proposed fully automated notarization often lump together several notarial systems, making it unclear whether they can be adapted to all types of notarizations.

Contribution. Therefore, we will focus only on fixed-date notarizations in this study and propose a system that can execute fixed dates without a third party (mainly a notary). Fixed-date notarizations authenticate the existence of a document, but not the authenticity of the content of the document. As such, it has potential for automation through smart contracts. Our proposed system eliminates the need for a verification authority to confirm the identity of the client by verifying signature data using a national eID card that implements Public Key Infrastructure (PKI) inside a public blockchain via smart contracts. Therefore, the verification meets the requirements of tamper resistance, high availability, and transparency of verification results.

Specifically, by performing verification within the smart contract on the electronic document signed using a national eID card with PKI, the signer is automatically confirmed, and the electronic document is authenticated. If the verification result is correct, the information is stored in the blockchain. The transaction receipt issued by the blockchain platform is used as proof of the notarization, thus creating a highly reliable system.

Additionally, this study presents a use case in Japan. In the case of Japan, notaries designated by the Minister of Justice performs the work manually, which takes time and effort. However, by replacing some of that work with smart contracts on the blockchain, the amount of time clients have to wait will be greatly reduced. Furthermore, we demonstrate the feasibility of this system for smartphone applications and the Ethereum blockchain.

The remainder of this paper is organized as follows: Section II describes some preliminaries, and Section III presents related studies. Section IV provides a detailed explanation of the proposed architecture. Section V presents the results of the implementation of the proposed architecture and its evaluation. Section VI presents a use case in Japan. Section VII presents the discussion. Finally, we conclude the paper in Section VIII.

II. PRELIMINARIES

A. BLOCKCHAIN

Blockchain is a distributed ledger technology invented by S. Nakamoto [8] and is the core technology of many cryptographic assets, including Bitcoin. Blockchain stores data by grouping transactions into blocks and embedding the hash value of one block into the next block that occurs on a peer-to-peer (P2P) network.

The nodes put together a transaction, generate a block, and are rewarded. This sequence of events is called mining. In particular, Bitcoin and Ethereum[1] employed the proof of work (PoW) method to determine the mining node, which is time-consuming. The main chain of the blockchain is the longest chain from the first block to the current block. Therefore, to falsify the data stored in a block, all subsequent blocks would have to be revoked, requiring many computations in the PoW. Such falsification is practically impossible.

Blockchain is tamper resistant. Additionally, public blockchains are highly transparent because anyone can participate in the network and see the data stored in the blocks.

B. SMART CONTRACT AND ETHEREUM

A smart contract is a program recorded on a blockchain and is falsifying resistant. When the program is executed, its record is stored in the blockchain, ensuring the transparency of the sequence of actions. Additionally, there is essentially no single point of failure as the program runs on a P2P network. Due to this characteristic, contracts can be executed automatically without a third party.

Ethereum [9], [10] is the first blockchain platform to adopt smart contracts, which are stored as Ethereum Virtual Machine (EVM) bytecodes. In this study, smart contracts were written in Solidity and compiled into EVM bytecodes using Remix-ide[2].

Ethereum uses the elliptic curve digital signature algorithm (ECDSA) [11], where the address of an account is created by taking a hash (keccak256 [12]) of the public key. This account is called an externally owned account (EOA), and the user executes a function in the smart contract through the EOA.

To execute a function in a smart contract, the EOA needs to issue a transaction, and a fee called gas is charged to the minor for that transaction. Generally, the more complex the processing within a function, the higher the gas value. Given the concept of gas, a large amount of gas would be required for a malicious attacker to attempt to perform a DoS attack that deploys computationally intensive transactions on the Ethereum network. Therefore, the attacker will lose the incentive to launch a DoS attack on Ethereum.

When a transaction is executed, a transaction receipt is issued summarizing execution results. The EOA that executes the transaction and the log of the execution is written in the transaction receipt and stored in each node.

C. NOTARIZATION AND FIXED DATES

Notarization is a system established by law to serve the public in general financial transactions, estates, deeds, powers of attorney, and non-litigious cases involving foreign and international business. The main roles of notarization, which vary among countries, are to authenticate the signatures of a person for the purpose of signing documents; administer oaths and affirmations; obtain affidavits from witnesses; authenticate the execution of certain types of documents; approve deeds and other assignments; notify foreign bills of exchange; and provide exemplary answers and notarized copies.

In this study, we focus on fixed dates. Fixed-date notarizations legally confirm the date of creation of a document and are used in some European countries and Japan. The reason this type of notarization exists is that it is often easy to fake the creation date of a document created by a private individual. Even a contract between two parties can be disguised as if it were a document created in the past by two people conspiring together. By using this system, it is possible to quickly prove the date of creation of a document in the event of a dispute, such as the assignment of a claim or a pledge of rights. However, the Officially-Attested Date is

[1]As of May 2022, Ethereum is transitioning from Proof of Work (PoW) to Proof of Stake (PoS).

[2]Remix, “Remix -Ethereum IDE," 2022, https://remix.ethereum.org/

only the confirmation of date; it does not certify matters such as the genuineness of the document creation.

D. NATIONAL EID CARD

A national eID card is an electronic identification card issued by the government. As of 2022, it has been adopted in Europe and many other countries [13]. Their main uses, which vary among countries, include facilitating the issuance of public transportation tickets, driver's licenses, and health insurance cards. Further, they also enable passport replacement within the EU, Internet voting, accessing citizen portals, and online banking. These are achieved by utilizing the digital signature function of the PKI recorded in the IC chip in the card.

The national eID card is based on ISO/IEC 7816, the standard contact type of smart card, or ISO/IEC 14443, the standard contactless type, with an Operation System (OS) corresponding to each card. Therefore, communication between the card reader and the card is executed via data packets called Application Protocol Data Units (APDUs). The card OS is responsible for analyzing the received APDUs and routing them to the appropriate applications. It is also responsible for collecting application responses and sending them to the card reader (or the application communicating with the reader).

When using the digital signature function of the ID card, a several-digit Personal Identification Number (PIN) must be entered from the application. Restrictions on the number of digits and character types vary depending on the country [13]. Therefore, assuming that only the legitimate cardholder knows the PIN and is in possession of the card, their identity can be verified electronically.

III. RELATED WORKS

A. LITERATURE REVIEW

Yermack [14] showed that blockchain technology has the potential to improve corporate governance and government services due to its various characteristics (low cost, high liquidity, more accurate record keeping, transparency, etc.). However, the study only describes how blockchain technology affects stakeholders and provides few technical descriptions.

Ølnes et al. [15] investigated whether blockchain technology can innovate and change governmental processes. The results indicate that a needs-driven approach, rather than a technology-driven approach, should be taken when applying blockchain to governmental processes. Additionally, they found that governments need two perspectives: governance through blockchain technology use and blockchain governance. The former is the use of blockchain technology in public organizations processes and blockchain use in governing their transactions. The latter is that governments determine how blockchain should look, and how to adapt to changes, and they should ensure that public values and societal needs are fulfilled. However, this study describes how governments should incorporate blockchain, and does not propose any technical solutions.

Geneiatakis et al. [16] examined whether cross-border e-government services could be a possible conversion destination for blockchain technology. They focused on SEED, a data-sharing platform for a system that monitors excise tax movements within EU member states and proposed and implemented a blockchain technology application to SEED. Specifically, they implemented their implementation using a Hyperledger Fabric (ver. 1.1) to meet the SEED requirements and demonstrated that the proposed method achieves a throughput of 8 transactions per second (tps) with a bandwidth of 4 Mb/s and 48 tps with a bandwidth of 1 Gb/s in a demonstration experiment with 28 nodes. However, since this study focuses on cross-border services in government processes, we cannot determine whether it can be applied to the notary system.

Páez et al. [4] proposed a blockchain-based biometric architecture for e-government. Specifically, the architecture creates a unique private blockchain consisting of nodes that perform biometric authentication and synchronize on a consensus algorithm proposed by the authors called Proof-of-Tournament. However, this study only implements the architecture, and it remains unclear whether it can be applied to government processes.

Gao et al. [5] focused on the unreliability and convenience of traditional manual notary publics and proposed a blockchain-based notary public architecture that improves on them. Specifically, they replaced some of the manual notary work on HyperLedger Fabric with smart contracts to guarantee high reliability. Their proposed method also supports cross-border use. Additionally, depending on the use, they divide the destination of the transaction into two cases: one to keep the transaction within the country and the other to deliver it to all nodes, which means that it crosses national borders. In this way, the throughput is comparatively high for in-country use, which is generally assumed to have many users. The writes throughput to the blockchain was 179 tps and 164 tps for the local and global ledgers, respectively.

Sousa et al. [6] proposed a microservices approach to blockchain technology that allows for integration between notary offices and other institutions and ensures security and speed in information exchanges between parties. As a prototype, they implemented a product that performs notarization procedures related to birth registration on the Ethereum blockchain. However, the only use of the blockchain in this study is certificate registration; the other programs are conventional programs implemented as microservices. Therefore, their proposed method as a product using smart contracts lacks transparency and availability.

Ulloa et al. [7] proposed a blockchain-based system as a replacement for conventional notarization. In a private blockchain involving notary publics and government organizations, the authors argue that smart contracts can be used to replace the cumbersome tasks in traditional notarization with programs. This system was implemented using the Ethereum private blockchain. However, the implemented system only records the notarization results on the blockchain through smart contracts, and we believe that it would be difficult to replace all notarization processes with this system.

B. DIFFERENCES OF THE PROPOSED METHOD FROM RELATED WORKS

There are many studies related to e-government blockchain, but few focus on the notary system, especially notary office work. Additionally, the studies [4]–[6] have partially replaced the systems currently used by notaries with blockchain and have not replaced all of the notary's work in that business. Moreover, while some studies [7] have proposed methods to completely replace notaries' work, it is unclear whether the proposed methods can be applied to all of the various notarization types.

Therefore, we propose a system that completely replaces the notary with a smart contract for the fixed date, among the many notarization types. The reason why we focused on the fixed date is that it authenticates the existence of a document, not the authenticity of its content. For a program to determine a document's content, it usually requires a lot of computation using machine learning and other methods. Additionally, the limitations of virtual machines make it difficult to implement machine learning in smart contracts. In other words, most notarizations require a judgment about the authenticity of the document's contents, which is often not possible on a smart contract. However, a fixed date is one of the few types of notarizations that does not require authentication of the document's content. In other words, it can be automated by smart contracts.

There are two main technical differences from other related works. The first is the use of public blockchains. It has higher tamper resistance and availability than private or consortium blockchains. Therefore, applications can be configured on them to inherit their characteristics. The second is the use of the PKI of the national eID card issued by the government. By doing so, it is possible to verify who issued the transaction in a highly anonymous public blockchain environment.

IV. PROPOSED ARCHITECTURE

A. OVERVIEW

We aim to realize a fully automated notarization system for fixed dates that satisfies the requirements of falsification resistance, high availability, and transparency of the verification results. Specifically, we propose a system that can prove the existence of a document and who signed it at a certain time by verifying signatures on electronic documents using a national eID card with PKI implemented and a smart contract on a public blockchain.

A PKI, which is a centralized system, adds real-world identities to the decentralized blockchain world. In public blockchains, accounts managed in private keys are often not tied to real-world personal information. That is, it is not possible to verify who, in the real world, issued the transaction. However, by verifying the signature of the PKI on the smart contract, the person who manages the PKI can verify who issued the transaction. In other words, the proposed method can confirm who issued the transaction, which is a necessary element of the fixed dates.

We assume that the government agency publishes the hash value of the public key of the national eID card on the smart contract. As a result, when the smart contract verifies the digital signature in the client's transaction, it is possible to verify that the public key is valid.

The following are the three main points of the proposed method.

- **Smart contract method**:
  As signature verification is performed on the smart contract, it is possible to verify the signature verification from the outside, which almost eliminates the possibility of system downtime. As a result, it is not necessary to assume a trust agency for verification.

- **Transaction receipts as proof**:
  When a client proves the creation date of a certain document to a verifier, it can be easily verified by submitting the transaction receipt to the verifier.

- **Identity verification using a combination of a national eID card and smart contracts**:
  The verification of the digital signature created with a national eID card is performed in a smart contract whose program is open to the public and cannot be tampered with. Therefore, only the person who possesses their national eID card and knows the PIN can create a digital signature and breakthrough verification.

B. COMPONENTS

- **Client**: A client has their national eID card issued by a government agency and can affix a digital signature to any electronic document by using a specific card reader. It is also possible to hash an electronic document using an application.

- **National eID card**: This card is issued by a government agency and held by the client. The client can obtain a digital certificate and create a digital signature using the PIN set by the client when the card is issued. In this study, we use the private key corresponding to the public key in the electronic certificate to create a digital signature for the bearer's signature.

- **Smart contract**: Smart contracts that are supposed to be managed by a government agency. There are two types of smart contracts: public key management smart contracts and signature verification smart contracts. The hash of the public key registered in a national eID card is recorded in the public key management smart contract. If it is not recorded, it indicates that the key is invalid. The signature verification smart contract implements the signature verification function, which verifies the digital signature to confirm that the sender owns their national eID card.

- **Government agency**: An organization that manages smart contracts and national eID cards with PKI implemented. It is responsible for issuing cards. It also manages key revocation by adding and deleting the public key of the card to the list of the public key management smart contracts.

- **Verifier**: An organization that verifies the existence of the document. They verify, from the transaction receipt and the document submitted by the client, that the document existed at the time the transaction was mined.

C. ATTACK MODEL

Assuming that national eID cards, smart contracts, and government agencies are trustworthy, we consider two attack models. There are three basic elements of fixed dates: "who," "when," and "which document." Therefore, we can assume that the attacker will attack the three elements in the following two ways:

- Impersonation: An attacker pretends to be someone and executes the fixed date.
- Falsification: An attacker falsifies a document with a fixed date that was executed in the past or falsifies the date of a specific document.

We discuss how to deal with these attacks in Section VII under Security Discussion.

D. SMART CONTRACT AND FUNCTION

There are two types of smart contracts: a signature verification smart contracts, which verify digital signatures, and a public key management smart contract, which manage public keys. Details of the implementation are described below.

1) Signature Verification Smart Contract

This smart contract verifies the digital signature and registers the hash value of the document in the blockchain and transaction receipt, which has two functions, **Register_Hash** and **Sig_Verify**. Its configuration is shown in Algorithm 1.

Register_Hash is a function typically executed by the client to register a hash of the document in the blockchain and receipt. This function first queries **Find-Key**, a function of the public key management smart contract, for the validity of the public key using an internal transaction to verify the revocation information. Then, only if the public key is valid, **Sig_Verify** is executed to perform signature verification. Then, if the signature is valid, the document hash, digital signature, and public key are registered in the blockchain as a set. The pseudo-code is shown in Algorithm 2.

Sig_Verify is the function that performs the signature verification and is executed by **Register_Hash**. The verification method (e.g., ECDSA, RSA, etc.) is expected to vary depending on the digital signature scheme used, however, it should follow the signature scheme of the national eID card. If the signature is correct, a response of "true" is returned. Otherwise, a response of "false" is returned. The pseudo-code is shown in Algorithm 3.

A flowchart of this smart contract is shown in Figure 1.

Algorithm 1 Signature Verification Smart Contract

```
contract Signature_Verification {
    // smartcontract interface
    Public_Key_Management IPKM;
    // variables
    struct set{
        bytes hash;
        bytes signature;
        bytes pub_key;
    }
    set[] Set;
    // event
    event Register(bytes, bytes, bytes);
    // functions
    function Register_Hash(bytes, bytes,
    bytes);
    function Sig_Verify(bytes, bytes,
    bytes) returns (bool);
    // constructor
    constructor (address) {
        IPKM = Public_Key_Management(address);
    }
}
```

Algorithm 2 Register_Hash

Input: pub_key[bytes], signature[bytes], hash[bytes]
Output: void

```
1: if IPKM.Find_key(pub_key) == true then
2:     if Sig_Verify(pub_key, hash, signature) == true then
3:         Set.push(set(pub_key, signature, hash));
4:         emit Register(pub_key, signature, hash);
5:     else
6:         revert;
7:     end if
8: else
9:     revert;
10: end if
```

Algorithm 3 Sig_Verify

Input: pub_key[bytes], signature[bytes], hash[bytes]
Output: bool

```
1: // Decryption methods vary depending on the signature
   protocol used
2: if decryption(signature, pub_key) == hash then
3:     return true;
4: else
5:     return false;
6: end if
```

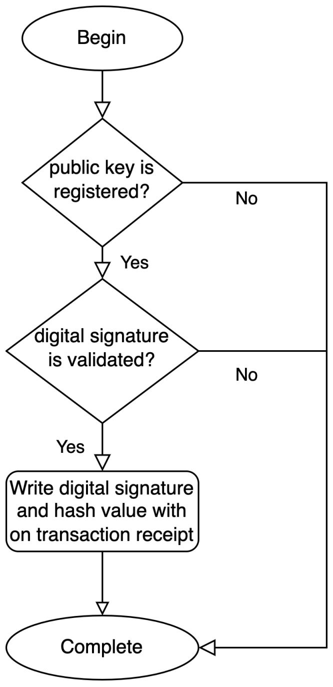

Begin
public key is registered?
No
Yes
digital signature is validated?
No
Yes
Write digital signature and hash value with on transaction receipt
Complete

FIGURE 1. Flowchart of Signature Verification Smart Contract

2) Public Key Management Smart Contract

This is a smart contract for government agencies to manage citizens' public keys. Its configuration is shown in Algorithm 4. This contract consists of three functions: the function **Add-Key**, which adds the hash value of the public key to the hash table, the function **Delete-Key** that deletes it, and the function **Find-Key** which returns true/false depending on if the hash value of the public key exists in the list. The pseudo codes are shown in Algorithms 5, 6, and 7, respectively. Note that **Add-Key** and **Delete-Key** are access-controlled so that only the EOA of the government agency can execute them.

Algorithm 4 Public Key Management Smart Contract

```
contract Public_Key_Management {
    // variables
    address owner;
    mapping(bytes => bool) hash_table;
    // functions
    function Add_key(bytes);
    function Delete_key(bytes);
    function Find_key(bytes);
    // constructor
    constructor (address) {
        owner == msg.sender;
    }
}
```

E. SYSTEM FLOW

The proposed method consists of four components. Figure 2 provides an outline.

Algorithm 5 Add-Key

```
Input: pub_key[bytes]
Output: void
1: if msg.sender == owner then
2:     hash_table[pub_key] = true;
3: else
4:     revert;
5: end if
```

Algorithm 6 Delete-Key

```
Input: pub_key[bytes]
Output: void
1: if msg.sender == owner then
2:     hash_table[pub_key] = false;
3: else
4:     revert;
5: end if
```

1) **Registering and deleting public keys**:
The government agency registers and deletes the client's public key using **Add-Key** and **Delete-Key** in the public key management smart contract.

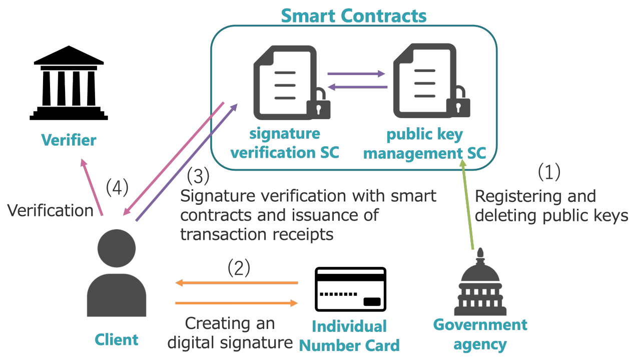
2) **Creating a digital signature**:
The client uses their national eID card and application to create a signature for the document's hash value.

3) **Signature verification with smart contracts and issuance of transaction receipts**:
By issuing a transaction that executes **Register_Hash**, the client sends the hash value of the document, the signature, and the public key used for the signature to the signature verification smart contract. The signature verification smart contract calls the **Find_Key** function of the public key management smart contract to confirm the validity of the public key. If it is valid, the signature is verified by calling **Sig_Verify** with the hash value of the electronic document, the signature, and the public key. If the verification result is correct, the hash of the document and the digital signature are written in the transaction receipt and issued. Fixed dates are completed when the transaction receipt is output and stored in the blockchain.

4) **Verification**:
The client submits the transaction receipt and the document itself to the verifier. The verifier computes a hash from the submitted document and compares it to the hash of the document entered in the transaction receipt. By doing so, the verifier can verify that the document existed when the transaction was issued. Therefore, the verifier can also confirm that the person who knows the PIN of their national eID card has created a digital signature.

Algorithm 7 Find-Key

```
Input: pub_key[bytes]
Output: bool
1: return hash_table[pub_key]
```



Smart Contracts
signature verification SC
public key management SC
Verifier
Verification
(4)
(3)
Signature verification with smart contracts and issuance of transaction receipts
(1)
Registering and deleting public keys
(2)
Client
Creating an digital signature
Individual Number Card
Government agency

FIGURE 2. Outline

In 3), if the transaction fails, it means that the National eID card is not registered in the Public Key Management Smart Contract, or that the document selected when creating the digital signature is different from the document selected when issuing the transaction. In the former case, it can be assumed that the national eID card has expired, so it is necessary to apply for a new card at the national office. For the latter case, it is necessary to check that the selected document is correct and then issue a new transaction. Additionally, if a transaction takes a long time to be approved, it may be necessary to raise the gas price.

V. USE CASE

In this section, we show how the proposed method can be adapted in Japan as a use case.

In Japan, Individual Number Cards are available as national eID cards, and they come equipped with digital signature functions compatible with the Japanese PKI (JPKI). Additionally, there is a system called Officially-Attesting Dates as a fixed date. These are described in detail below.

A. JPKI AND INDIVIDUAL NUMBER CARD

JPKI is a framework for identity verification when performing administrative procedures online in Japan and requires a digital signature by the Japanese individual number card [17]. The Japan Agency for Local Authority Information Systems (J-LIS) issues cards. An individual number card contains two types of digital certificates: **an electronic certificate for the bearer's signature** and **an electronic certificate for user identification**. The four basic information[3] includes only an electronic certificate for the bearer's signature. Additionally, due to the revision of the Public Personal Identification Law, the private sector has been able to use this system since January 2016. Private businesses that have obtained permission

[3]bearer's name, address, date of birth, and sex

TABLE 1. Card Application

Card Application	Usage
JPKI-AP	Application that performs user authentication and digital signatures
Ticket Surface Information Confirm AP	Application holding the information on the card face as image data
Ticket Surface Information Input AP	Application holding the information on the card face as text data
Residential Basic Book AP	Application holding the civil registration code
Other AP	Empty space Local governments can include their applications in this space

from the Minister of Internal Affairs and Communications can perform user authentication using the individual number card. As of October 2021, 14 companies were certified by the Minister of Internal Affairs and Communications, and 113 private companies used their systems to provide certification services.

The specifications of the Japanese individual number card were not officially released. An individual number card is an eID card that has both contact and contactless interfaces. The contact interface conforms to ISO/IEC 7816 and can be used with general contact IC card readers. The contactless interface conforms to ISO/IEC 14443 Type B and can be used with near-field communication (NFC)-enabled IC card readers.

The IC chip of the individual number card is equipped with an operating system for the eID card, which consists of four card applications. Table 1 lists their types and applications. The electronic certificate for the bearer's signature and the electronic certificate for user identification is provided in JPKI-AP; it is necessary to use the 4-digit number for the former and 6~16-digit alphanumeric password (PIN) for the latter set at the time of issuance of the individual number card to obtain them. Additionally, any electronic document can be digitally signed with the public key listed in the certificate and the corresponding private key. However, the card will be locked if the PIN is entered incorrectly three times for the electronic certificate for user identification and five times for the electronic certificate for the bearer's signature. Because the lock can only be unlocked at the local government office, the system is resistant to password attacks.

B. OFFICIALLY-ATTESTING DATES

In Japan, the system whereby a notary public stamps a document to legally certify that the document existed on that date is called Officially-Attesting Dates [18]. This system does not authenticate the truthfulness of the contents of the document; therefore, the notary does not verify the contents of the document.

In the case of paper documents, the procedure is completed by a notary designated by the Minister of Justice, who stamps the document with his seal at the notary office. In the case of electronic documents, the notary completes the process by affixing an electronic signature at the notary office. Additionally, if the client wishes, electronic documents with

Officially-Attesting Dates can be stored for 20 years and related data for 50 years on the notary public's server.

The following is a description of the current procedure in Japan for attaching the Officially-Attesting Dates to an electronic document.

1) The client obtains an electronic certificate from the Japanese individual number card.
2) The client creates the document as an electronic file (only PDF is acceptable).
3) The client contacts notary by phone or fax.
4) The client affixes a digital signature to the created electronic file.
5) The client uses the online application system provided by the Ministry of Justice and sends the electronic certificate and electronic file to the notary.
6) The client goes to the notary public's office, and the notary confirms that the electronic signature has been made in front of the client.
7) If there are no problems as a result of the review by the notary, the notary will affix a digital signature to the electronic file upon payment of a fee by the client.
8) The client receives the electronic file with the notary's electronic signature on an electronic medium (e.g., USB or CD) brought by the client [4].

There are three requirements for Officially-Attesting Dates [18].

1) It must be a private document duly prepared.
2) It must be a meaningful document.
3) It must contain the signature or the name-seal of the person who prepared it.

C. HOW THE PROPOSED METHOD ADAPTS

By using the proposed method, Officially-Attesting Dates that is the fixed dates in Japan can be realized as a fully automated system that does not require a notary public. Specifically, digital signature with an individual number card proves the identity of the person signing the document. The important point is that the verification process is performed on the smart contract. In this way, the verification by the notary can be replaced by a secure and highly available program, eliminating mistakes and fraud conducted by the notary systems.

Additionally, in the proposed method, the requirements of Officially-Attesting Dates for **1)** and **2)** are satisfied because the verifier checks the document after its submission to the client. The requirement for **3)** is also satisfied because the document is digitally signed with a Japanese individual number card. Therefore, we can say that the proposed method satisfies the requirement of Officially-Attesting Dates in Japan.

VI. IMPLEMENTATION AND EVALUATION

In this study, as a prototype of the proposed method, we implement a digital signature application using a Japanese individual number card and smart contracts to verify the digital signature. For the former, we used Swift to create an application that runs on a smartphone running on IOS using Xcode 12.5 and demonstrated it on an iPhone Xs running on IOS 14.6. For the latter, we used Solidity to create a smart contract that works on Rinkeby, one of the Ethereum test networks, using Solidity version 0.6.0.

[4]Receiving the file via the Internet is not permitted

A. DIGITAL SIGNATURE APPLICATION USING AN INDIVIDUAL NUMBER CARD

We will use Swift to create an application that runs on the IOS. There are three main functions; the implementation details are described below. Additionally, sequence and outline diagrams are shown in Figures 3 and 4, respectively.

1) Obtaining an electronic certificate for the bearer's signature from an individual number card

As the Japanese individual number card conforms to ISO7816, communication between the card and the application is possible by using the APDU command. In Swift, the **coreNFC** framework can be used to send APDU commands to the eID card via the NFC.

Figure 5 shows the configuration of JPKI-AP. An electronic certificate for the bearer's signature can be obtained by sending the APDU command to the individual number card according to the following procedure:

1) **SELECT FILE** JPKI-AP
2) **SELECT FILE** PIN for the bearer's signature
3) **VERIFY** PIN for the bearer's signature[5][6]
4) **SELECT FILE** Electronic certificate for the bearer's signature
5) **READ BINARY**

The electronic certificate for the bearer's signature obtained in the Distinguished Encoding Rules (DER) format is stored in a secure area inside the application. The pseudo-code for this function is shown in Algorithm 8.

Algorithm 8 get_certificate

```
Input: PIN
Output: certificate
1: // sendcommand means sending an APUD command to the card.
2: sendcommand(SELECT FILE <JPKI-AP>)
3: sendcommand(SELECT FILE <PIN for the bearer's signature>)
4: sendcommand(VERIFY(PIN))
5: sendcommand(SELECT FILE <Electronic certificate for the bearer's signature>)
6: certificate = sendcommand(READ BINARY)
7: write(certificate)
```

[5]The 6~16 digit alphanumeric code that you set when you received your Japanese individual number card

[6]The electronic certificate for the bearer's signature can be accessed by updating the security status with the verification of the PIN

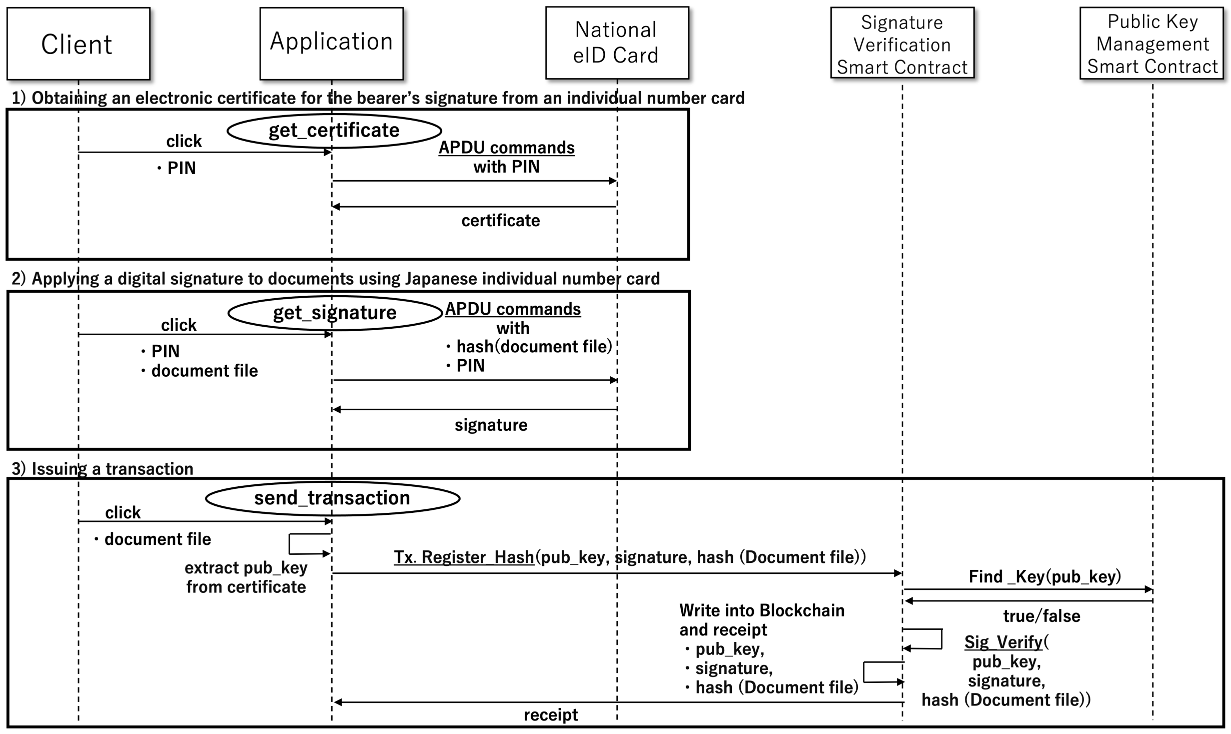

Client
Application
National eID Card
Signature Verification Smart Contract
Public Key Management Smart Contract
1) Obtaining an electronic certificate for the bearer's signature from an individual number card
get_certificate
click
・PIN
APDU commands with PIN
certificate
2) Applying a digital signature to documents using Japanese individual number card
get_signature
click
・PIN
・document file
APDU commands with
・hash(document file)
・PIN
signature
3) Issuing a transaction
send_transaction
click
・document file
extract pub_key from certificate
Tx. Register_Hash(pub_key, signature, hash (Document file))
Find _Key(pub_key)
true/false
Write into Blockchain and receipt
・pub_key,
・signature,
・hash (Document file)
Sig_Verify(pub_key, signature, hash (Document file))
receipt

FIGURE 3. Sequence diagram

2) Applying a digital signature to documents using Japanese individual number card

The first step is to obtain the hash value of the document. The next step is to create a digital signature of the hash value of the document using the APDU command, as in 6.1.1. By sending the APDU command to the individual number card using the procedure shown below, this function creates a digital signature using the secret key in the card.

1) **SELECT FILE** JPKI-AP
2) **SELECT FILE** PIN for the bearer's signature
3) **VERIFY** PIN for the bearer's signature[7]
4) **SELECT FILE** Private key for the bearer's signature
5) **COMPUTE DIGITAL SIGNATURE**

The digital signature is stored in a secure area inside the application. The pseudo-code for this function is shown in Algorithm 9.

3) Issuing a transaction

This function extracts the public key from the electronic certificate for the bearer's signature obtained in **1)** and issues a transaction, which is sent the key to the smart contract along with the digital signature obtained in **2)** and the hash value of the document. In this case, we use Ethereuem's node hosting service, called Infura[8], to connect to the test network Rinkeby. Then, when the smart contract successfully verifies the transaction, this application receives the transaction receipt and stores it in a secure area. The pseudo-code for this function is shown in Algorithm 10.

Algorithm 9 get_signature

```
Input: PIN, Document file
Output: signature
1: document_hash = sha256(Document file)
2: sendcommand(SELECT FILE <JPKI-AP>)
3: sendcommand(SELECT FILE <PIN for the bearer's signature>)
4: sendcommand(VERIFY(PIN))
5: sendcommand(SELECT FILE <Private key for the bearer's signature>)
6: signature = sendcommand(COMPUTE DIGITAL SIGNATURE(document_hash))
7: write(signature)
```

B. SMART CONTRACT

We used Solidity to create smart contracts that ran on Ethereum. The composition of the smart contract is basically as shown in Section IV. However, because the signature method of the Japanese individual number cards is 2048-bit RSA, the verification method is implemented in the **Sig_Verify** function. In this implementation, we use PKCS#1 SHA256 as the padding and implement RSA signature verification in a function in a smart contract.

However, for 2048-bit RSA signatures, the amount of computation required to verify the signature is quite large,

[7]The same PIN as *5
[8]ConsenSys, Infura, https://infura.io/, 2022

Algorithm 10 send_transaction

Input: Document file
Output: receipt

```
1: document_hash = sha256(Document file)
2: // extract the public key from the certificate
3: pub_key = get_pubkey(certificate)
4: // send a transaction that executes Register_Hash to the
   address of the smart contract that is hard-coded
5: receipt = sendTransaction(Regsiter_Hash(pub_key, signature, document_hash))
6: write(receipt)
```

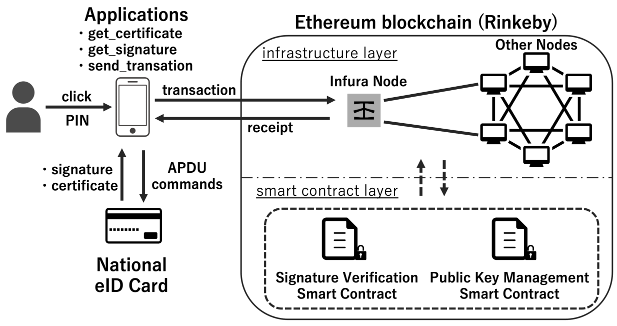


FIGURE 4. Outline of prototype

and the amount of gas required to run a smart contract on Ethereum would be enormous. From the client's financial point of view, it is not the intention to lose a large asset simply by applying a digital signature.

Therefore, in this implementation, we attempt to significantly reduce the gas for RSA signature verification using EIP-198 [19], which was implemented in Ethereum. EIP-198 efficiently calculates modulo exponentiation and reduces the amount of gas generated in the process. Therefore, a smart contract is not filled with a program that calculates modulo exponentiation but with a program that calls EIP-198. For the implementation, we used publicly available, open-source code [20].

C. EVALUATION OF FEASIBILITY

We show that the prototype implementation works.

First, from the government agency's standpoint, we add the client's public key to the list on the public key management smart contract using the function **Add-Key**. Figure 6 shows the transaction.

Second, from the client's standpoint, we use the application to create a digital signature from the individual number card and use the function **Register_Hash** to verify the signature in the smart contract. Figure 7 shows the transaction. This result shows that the application was able to create a signature from the individual number card and verify it within the smart contract because the private key in the card cannot be retrieved even by the client.

Finally, we present in Table 2 the processing times for the three implemented applications. These measurements

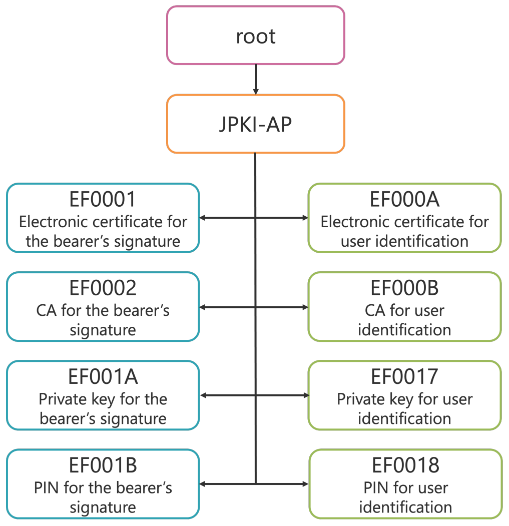

root
JPKI-AP
EF0001
Electronic certificate for the bearer's signature
EF000A
Electronic certificate for user identification
EF0002
CA for the bearer's signature
EF000B
CA for user identification
EF001A
Private key for the bearer's signature
EF0017
Private key for user identification
EF001B
PIN for the bearer's signature
EF0018
PIN for user identification

FIGURE 5. Directory diagram of JPKI-AP

are averages of 100 runs of each application. The reason why there is a difference of more than 6 seconds between get_certificate and get_signature may be that get_signature creates digital signatures in a card with lower processing power than a normal PC. Additionally, it is important to note that the measurement of the get_transaction is taken on Rinkeby, where tps is a theoretical maximum of 20 tps. Moreover, this value is measured when mining is performed within 1 block of the inclusion time in the pending pool.

TABLE 2. Processing Time

Application	processing time
get_certificate	1.333 seconds
get_signature	7.555 seconds
send_transation	8.787 seconds

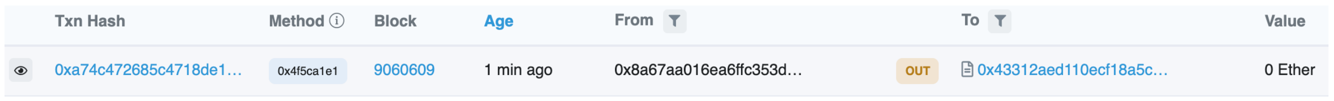
VII. DISCUSSION

A. COMPARISON WITH OTHER ARCHITECTURES

A comparison showing studies that would replace notary work is shown in Table 4. The primary reason for comparing our proposed method with these studies is the similarity it shares with our research aim, which is to conduct notary work using blockchain technology.

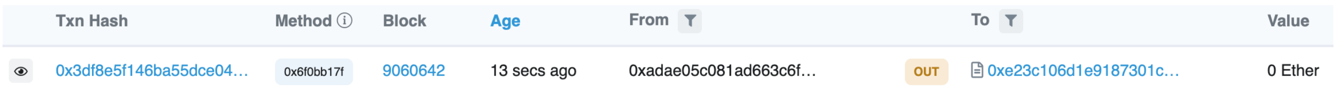
The gas fee, which is a blockchain fee, is not charged by the methods of Páez et al. and Gao et al. because they use the original chain and Hyperledger Fabric, respectively. However, the other proposed methods using Ethereum, including ours, charge a gas fee.

Additionally, while other studies aim to assist notaries in their work by configuring blockchain systems, ours aims to replace the notary's work with smart contracts. Therefore, our architecture has a narrower scope than others. It only covers fixed dates that can be executed by smart contracts because it does not require confirmation of the document's content. Although the method proposed by Ulloa et al. has



Txn Hash	Method ⓘ	Block	Age	From		To	Value
0xa74c472685c4718de1…	0x4f5ca1e1	9060609	1 min ago	0x8a67aa016ea6ffc353d…	OUT	0x43312aed110ecf18a5c…	0 Ether

FIGURE 6. Transaction from government agency (Add-key)



Txn Hash	Method ⓘ	Block	Age	From		To	Value
0x3df8e5f146ba55dce04…	0x6f0bb17f	9060642	13 secs ago	0xadae05c081ad663c6f…	OUT	0xe23c106d1e9187301c…	0 Ether

FIGURE 7. Transaction from client (Register_Hash)

TABLE 3. Address

Components	Address
Client EOA	0xAdae05c081Ad663C6f0106e6B32b0728771dFcB3
Government agency EOA	0x8a67AA016EA6fFC353D6ea236A3141EB73cebD2B
Signature verification smart contract	0xE23C106D1E9187301c72135d5C4bE7D7C4e138fA
Public key management smart contract	0x43312aEd110eCf18A5C3168531067F70e4141F8d

wide coverage in notarial work and claims to be fully replaceable, it is unlikely to replace everything in notarial work, where document content verification is often mandatory.

In terms of throughput, it is obvious that the methods of Páez et al. and Gao et al. using original chains and consortium-type blockchains have a high throughput. Our proposed method depends on the throughput of the public blockchain used. In this case, we used Rinkeby, one of Ethereum's testnets, which has a throughput of about 20 tps. However, higher throughput is expected through the use of Ethereum's layer 2 solution and the use of high throughput public blockchains.

TABLE 4. Comparison with other architectures

Architecture	Blockchain	gas fee	scope	fully replacement	throughput (tps)
Páez et al. [4]	Original chain	No	wide	No	1886
Gao et al. [5]	Hyperledger Fabric	No	wide	No	164~174
Sousa et al. [6]	Ethereum (private)	Yes	wide	No	-
Ulloa et al. [7]	Ethereum (private)	Yes	wide	Yes	-
Our architecture	Ethereum (public)	Yes	narrow	Yes	20

B. RELATIONSHIP BETWEEN A NATIONAL EID CARD AND EOA

In the proposed method, there is no need to link a national eID card to the EOA. In other words, a transaction can be issued using any EOA, provided that the client can prepare their national eID card and the PIN of the electronic certificate for the bearer's signature entered at the time of issuance. Therefore, the proposed method has the advantage that there is little risk of the private key of the EOA being stolen. Hence, there is no need for the strict management of the private key of the EOA in our system.

Additionally, the tamper resistance of a document until it is stored in the blockchain depends only on the RSA signature of the national eID card. Even if a transaction is issued with a fake digital signature, the transaction can be rejected by verifying it with a signature verification smart contract.

C. TRANSACTION RECEIPT

This section discusses how transaction receipts can be used as proof of an official attesting date. In the past, digital signatures have been used to show that a document has not been tampered with and the issuer of the document has indeed issued it. In this study, however, we use transaction receipts instead of digital signatures. Therefore, it is necessary to show that this receipt has not been tampered with and is issued correctly. The following two points show that it is difficult to tamper with the receipt.

1) The first point is about tamper resistance before the blockchain is stored. In order to tamper with receipts generated from transactions, Ethereum clients, such as Geth and Parity, managed by OSS, need to be tampered with. However, it is practically impossible to modify the nodes of all participants in a blockchain. Additionally, even if a block containing a tampered receipt is propagated to other nodes, the other nodes will not receive the block and thus cannot spread the tampered receipt. Furthermore, it is also important that the smart contract that issues the receipt is tamper resistance.
2) The second point is about the tamper resistance after the blockchain is stored. Because the receipts stored in the blockchain are protected by a cryptographic hash function, this tampering is practically impossible.

Therefore, from the above two points, we can say that transaction receipts are tamper resistant.

D. DIVIDING THE SMART CONTRACT INTO TWO PARTS

It is technically feasible to combine the functions of a signature verification smart contract and a public key management smart contract and operate them as a single smart contract. However, in this study, the reason for dividing them into two smart contracts is that there may be multiple smart contracts for each. For example, there may be multiple public key management smart contracts for each local government, or multiple signature verification smart contracts with different signature schemes and uses accessing the public key management smart contract. Considering this possibility, we divided the functions into two smart contracts, one for public key management and the other for signature verification, because we thought that dividing the functions would increase the extendability of the proposed method.

E. SECURITY DISCUSSION

We discuss the security of the proposed system under the assumption that RSA signatures, hash functions, blockchains, and national eID cards are secure. The possible attacks on our proposed system can be classified into the following two patterns.

1) Impersonation

For the attacker to succeed in this attack, they need to obtain the private key in the national eID card. There are two possible ways for an attacker to steal the private key: stealing the national eID card and illegally extracting the RSA signatures or public keys.

However, under the assumption that the national eID card and RSA signature are secure, the attack will fail. This is because even if the attacker manages to steal a national eID card, he cannot create a digital signature unless he knows the PIN registered by the owner when the card was issued. Additionally, National eID cards are tamper resistant, with the card's circuitry designed to break when an attacker attempts to physically analyze it. It also has a safety feature that locks the card if the PIN input fails several times. At that point, only specific organizations can unlock it. Therefore, brute force attacks cannot be used. For the latter, it is easy for an attacker to obtain RSA signatures and public keys because they are publicly available. However, under the assumption that RSA signatures are secure, it is not possible to extract private keys from them.

2) Falsification

For the attacker to succeed in this attack, they need to tamper with the hash of the document recorded in the blockchain. However, under the assumption that both the hash function and blockchain platform are sufficiently tamper resistant, meaning that the blockchain has a high enough block height and a large number of participating validator nodes, it is almost impossible to falsify the hash once it is recorded in the blockchain.

Formal proofs, together with formal security definitions, are planned in our future work.

F. WHY DOES OUR PROPOSED METHOD USE A PUBLIC BLOCKCHAIN?

There are two reasons for adopting a public blockchain for our proposed method.

1) First, public blockchains have higher tamper resistance and availability than consortium and private blockchains. We believe that these two factors are also important for the fixed date. Generally, it depends on the consensus algorithm. The greater the number of nodes participating in a blockchain, the higher the availability and tamper resistance. Therefore, public blockchains are more secure than other types because they have a larger number of participating nodes and are more geographically dispersed.
2) Second, public blockchains have more transparency than consortium and private blockchains. In public blockchains, the source code is published as a smart contract, and the client can verify that the code is working. Therefore, they can verify their trust in the system themselves. However, in the consortium and private blockchains, the source code is often not publicly available. This means that clients must trust the organization that manages the blockchain to use the system.

Generally, public blockchains also have the disadvantages of charging fees for gas and low scalability. However, the former is less economically burdensome for clients, because fees are also charged for traditional fixed dates, and less expensive public blockchains are emerging. For the latter, we believe that the influence is small because citizens do not frequently use notarization. Additionally, the scalability problem of public blockchains is being solved with the advent of Ethereum's layer 2 solutions, such as the optimistic-Rollup and zk-Rollup, and the high throughput blockchain called "Ethereum-Killers." Therefore, in the future, as digital society evolves, we believe that these solutions will be able to address the increasing frequency of fixed dates.

Additionally, the reason why we use Ethereum for the prototype implementation among the many public blockchains is that the most necessary element for fixed dates is tamper resistance, and we believe Ethereum best meets this requirement among blockchains that can run smart contracts. This is because Ethereum has more nodes participating as validators than any other chain and has a history of operating relatively securely among the many public blockchains.

G. LIMITATIONS OF OUR PROPOSED METHOD

There are two limitations of the proposed method that we can consider.

1) First, our proposed method is not adaptable to all notary systems, but to fixed dates only. This is because tasks such as verifying the content or authenticity of a document cannot generally be realized on a smart contract. In the case of fixed dates, there is no need to check the authenticity of the content, so it is feasible.

2) Second, the document itself cannot be preserved. In the case of a fixed date at a notary office, the notary can store a copy of the document. However, the proposed method only places a hash of the document on the blockchain. Therefore, if the document itself is lost, its management is left to the customer as the existence of the document cannot be verified. Nonetheless, we believe that this problem can be easily solved by stor-

ing encrypted documents in distributed storage, such as InterPlanetary File System (IPFS) [21].

H. POTENTIAL APPLICATIONS OF OUR PROPOSED METHOD

The proposed method performs PKI verification of the national eID card on a public blockchain and records a hash in the blockchain if the verification is valid. Therefore, the proposed method has the potential to be applied to fields other than notarization. For example, the proposed method can be applied to electronic medical records. It is possible to verify who created the medical record, when the record was created, and if the record has been tampered with. The proposed method could also be applied to intellectual property and legal documents.

Additionally, documents registered by the proposed method will be useful for cybercrime investigation and forensics, because who and when the transaction was issued will be recorded on the public blockchain.

VIII. CONCLUSION

In this study, we proposed an automatic notarization system for fixed-date notarizations, which does not require a third party, such as a notary public. The proposed system combines a digital signature function using a national eID card with PKI implemented and automatic contract execution using smart contracts. It uses transaction receipts as certificates of fixed-dates. As a use case, we introduced a fixed-date system in Japan and showed the feasibility of our proposed method using an individual number card (Japanese national eID card) and the Ethereum blockchain. In future work, we plan to improve the architecture to save the document because the proposed method leaves it to the client to save the document.

IX. ACKNOWLEDGMENTS

This work was partly supported by GMO Internet Foundation and Grant-in-Aid for Scientific Research (B) (22H03588).

REFERENCES

[1] Aji Supriyanto and Khabib Mustofa. E-gov readiness assessment to determine e-government maturity phase. In 2016 2nd International Conference on Science in Information Technology (ICSITech), pages 270–275. IEEE, 2016.
[2] Andreas Poller, Ulrich Waldmann, Sven Vowé, and Sven Türpe. Electronic identity cards for user authentication-promise and practice. IEEE Security & Privacy Magazine, 10(1):46–54, 2012.
[3] Gazmend Krasniqi and Kristaq Filipi. Efficiency comparison of cryptographic applications, match-off-card vs. match-on-card, using national biometric eid card. Academic Journal of Interdisciplinary Studies, 8(1):77–77, 2019.
[4] Rafael Páez, Manuel Pérez, Gustavo Ramírez, Juan Montes, and Lucas Bouvarel. An architecture for biometric electronic identification document system based on blockchain. Future Internet, 12(1):10, 2020.
[5] Ying Gao, Qiaofeng Pan, Yangliang Liu, Hongliang Lin, Yijian Chen, and Quansi Wen. The notarial office in e-government: a blockchain-based solution. IEEE Access, 9:44411–44425, 2021.
[6] Pamella Soares de Sousa, Nataniel Parente Nogueira, Rayane Celestino dos Santos, Paulo Henrique M Maia, and Jerffeson Teixeira de Souza. Building a prototype based on microservices and blockchain technologies for notary's office: An academic experience report. In 2020 IEEE International Conference on Software Architecture Companion (ICSA-C), pages 122–129. IEEE, 2020.
[7] Roberth Ulloa and Pablo Gallegos. Design of a blockchain architecture and use of smart contracts to improve processes in notary office. In International Conference on Applied Technologies, pages 469–483. Springer, 2021.
[8] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review, page 21260, 2008.
[9] Ethereum. Ethereum whitepaper. https://ethereum.org/en/whitepaper/, 2022.
[10] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper, 151(2014):1–32, 2014.
[11] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). International journal of information security, 1(1):36–63, 2001.
[12] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak. In Annual international conference on the theory and applications of cryptographic techniques, pages 313–314. Springer, 2013.
[13] Abubakar-sadiq Shehu, António Pinto, and Manuel E Correia. On the interoperability of european national identity cards. In International Symposium on Ambient Intelligence, pages 338–348. Springer, 2018.
[14] David Yermack. Corporate governance and blockchains. Review of finance, 21(1):7–31, 2017.
[15] Svein Ølnes, Jolien Ubacht, and Marijn Janssen. Blockchain in government: Benefits and implications of distributed ledger technology for information sharing, 2017.
[16] Dimitris Geneiatakis, Yannis Soupionis, Gary Steri, Ioannis Kounelis, Ricardo Neisse, and Igor Nai-Fovino. Blockchain performance analysis for supporting cross-border e-government services. IEEE Transactions on Engineering Management, 67(4):1310–1322, 2020.
[17] J-LIS. About an individual card. https://www.kojinbango-card.go.jp/en-kojinbango/, 2021.
[18] Japan National Notaries Association. How to make good use of japanese notaries. https://www.koshonin.gr.jp/pdf/english2.pdf, 2021.
[19] Ethereum. Eip-198: Big integer modular exponentiation. https://eips.ethereum.org/EIPS/eip-198, 2017.
[20] adria0. Solrsaverify. https://github.com/adria0/SolRsaVerify, 2017.
[21] Juan Benet. Ipfs-content addressed, versioned, p2p file system. arXiv preprint arXiv:1407.3561, 2014.

SHINYA HAGA received the B.S. degree in information science from University of Tsukuba, Japan in 2021 and is a student of master's program in Risk and Resilience Engineering in University of Tsukuba, as of 2022. His research interest includes social implementation of blockchain technology. He received the President's Award of the University of Tsukuba in 2021.

KAZUMASA OMOTE received his Ph.D. degrees in information science from Japan Advanced Institute of Science and Technology (JAIST) in 2002. He worked at Fujitsu Laboratories, LTD in 2002-2008, and was engaged in research and development for network security. He was a research assistant professor at JAIST since 2008, and associate professor at JAIST since 2011. He has been an associate professor at University of Tsukuba since 2016. His research interests include applied cryptography, network security and blockchain security. He was General Co-Chair of International Conference ACNS 2021. He received the WISTP 2019 Best Paper Award.