METHODS AND DEVICES FOR PROVIDING DECENTRALIZED IDENTITY VERIFICATION

TECHNICAL FIELD

[0001] The specification relates generally to computer technologies, and more particularly, to methods and devices for providing decentralized identity verification.

BACKGROUND

[0002] Traditional identity verification may use physical documents such as government-issued identity cards, passports, and the like to identify users. These documents, however, are not designed to be shared electronically, and sharing information contained in such documents may expose the users to potential harms such as identity theft and the like.

[0003] Decentralized identifiers (DIDs) are identifiers that users can create and manage on their own. DIDs may reduce the need for centralized authorities in identity management, and they may allow users to retain control over their own identifiers. Current implementations of DIDs include utilizations of DIDs in decentralized networks. Such decentralized networks may include blockchain systems.

[0004] Blockchain systems, also known as distributed ledger systems (DLSs) or consensus systems, may enable participating entities to store data securely and immutably. Blockchain systems may include any DLSs, without referencing any particular use case, and may be used for public, private, and consortium blockchain networks. A public blockchain network is open for all entities to use the system and participate in the consensus process. A private blockchain network is provided for a

particular entity, which centrally controls read and write permissions. A consortium blockchain network is provided for a select group of entities, which control the consensus process, and includes an access control layer.

[0005] Users of a blockchain system may include, e.g., individuals, businesses, banks, financial institutions, hospitals, as well as other types of companies, organizations, and the like. Some of these users may create DIDs and attempt to use their DIDs to prove their identities to others. For example, a first user, referred to as a DID Owner, may create a DID that contains certain identifiable information. The identifiable information may include, e.g., name, address, phone number, email address, as well as other types of information the DID Owner wants to use as a part of the DID. Once the DID is created, the DID Owner may attempt to obtain some type of confirmation from a second user, referred to as an Issuer, who is willing to issue a confirmation document digitally signed by the Issuer (e.g., using the Issuer's private key) confirming the identity of the DID Owner. The Issuer may be willing to issue such a confirmation document because the Issuer is in a certain type of relationship with the DID Owner. For example, the DID Owner may use an e-commerce platform provided by the Issuer, in which case the Issuer may be willing to issue a confirmation document confirming the identity of the DID Owner when requested by the DID Owner. In another example, the DID Owner may be an employee of the Issuer, in which case the Issuer may be willing to issue a confirmation document confirming the employment of the DID Owner.

[0006] After the Issuer issues the confirmation document confirming the identity of the DID Owner, the DID Owner may encrypt the confirmation document, e.g., using a cryptographic hash, and record a copy of the encrypted confirmation document on the blockchain system. The DID Owner may then provide the

unencrypted confirmation document to a third user, referred to as a Relying User, in an attempt to prove the identity of the DID Owner to the Relying User. The Relying User may be, e.g., a service provider or a bank who wants to verify the identity of the DID Owner prior to rendering a service or issuing a loan to the DID Owner. After verification, the Relying User may accept the identity claimed by the DID Owner as true.

[0007] While utilizing such DIDs may allow users to retain control over their own identifiers, doing so may require the DID Owners to have the abilities to store and manage confirmation documents they receive from various Issuers. For example, if a DID Owner has requested multiple confirmation documents from multiple Issuers, the DID Owner may be responsible for storing and managing multiple confirmation documents. Managing multiple confirmation documents may become complicated and time-consuming.

SUMMARY

[0008] In one aspect, a computer-implemented method for providing decentralized identity verification includes: recording an encrypted decentralized identifier (DID) received from a first user; recording a request for confirmation of truthfulness of the encrypted DID, the request being received from a second user; receiving from a third user a response to the request for confirmation and a proof for proving the third user is qualified to respond to the request for confirmation; recording the response to the request for confirmation; and in response to a determination that the proof is acceptable, indicating the response as valid.

[0009] In another aspect, a device for providing decentralized identity verification includes: one or more processors; and one or more computer-readable

memories coupled to the one or more processors and having instructions stored thereon that are executable by the one or more processors to: record an encrypted decentralized identifier (DID) received from a first user; record a request for confirmation of truthfulness of the encrypted DID, the request being received from a second user; receive from a third user a response to the request for confirmation and a proof for proving the third user is qualified to respond to the request for confirmation; record the response to the request for confirmation; and in response to a determination that the proof is acceptable, indicate the response as valid.

[0010] In still another aspect, a non-transitory computer-readable medium has stored therein instructions that, when executed by a processor of a device, cause the device to perform a method for providing decentralized identity verification. The method includes: recording an encrypted decentralized identifier (DID) received from a first user; recording a request for confirmation of truthfulness of the encrypted DID, the request being received from a second user; receiving from a third user a response to the request for confirmation and a proof for proving the third user is qualified to respond to the request for confirmation; recording the response to the request for confirmation; and in response to a determination that the proof is acceptable, indicating the response as valid.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate embodiments. In the following description, which refers to the drawings, the same numbers in different drawings represent the same or similar elements unless otherwise represented.

[0012] FIG. 1 is a schematic diagram of a blockchain system, according to an embodiment.

[0013] FIG. 2 is a schematic diagram of a computing device for implementing a node in a blockchain system, according to an embodiment.

[0014] FIG. 3 is a flow chart of a method for providing decentralized identity verification, according an embodiment.

[0015] FIG. 4 is a flow chart of a method for providing decentralized identity verification, according an embodiment.

[0016] FIG. 5 is a block diagram of an apparatus for providing decentralized identity verification, according to an embodiment.

DETAILED DESCRIPTION

[0017] Embodiments of the specification provide methods and devices for providing decentralized identity verification. The methods and devices may allow a first user of a blockchain system, referred to as a DID Owner, to define its own decentralized identifier (DID) and record an encrypted (e.g., hashed) version of the DID on the blockchain system. The methods and devices may also allow the DID Owner to present the DID to a second user of the blockchain system, referred to as a Relying User, who may verify the validity of the DID presented by the DID Owner based on the encrypted DID recorded on the blockchain system. The methods and devices may further allow the Relying User to record a request on the blockchain system requesting for confirmation of truthfulness of the encrypted DID recorded on the blockchain system. The methods and devices may allow a third user of the blockchain system, referred to as a Responder, to respond to the recorded request along with a proof indicating that the Responder is qualified to respond to the

recorded request. In this manner, the methods and devices can support DID verification without requiring an Issuer to issue any confirmation document, which in turn eliminates the need for the DID Owner to store and manage the confirmation document.

[0018] Embodiments disclosed in the specification have one or more technical effects. In some embodiments, the methods and devices support recordation of encrypted DIDs on a blockchain system. This allows the methods and devices to facilitate decentralized identity verification using the blockchain system and allows Relying Users to verify the validities of DIDs based on the encrypted DIDs recorded on the blockchain system. In some embodiments, the methods and devices support recordation of requests on the blockchain system. This allows the methods and devices to provide a mechanism for the Relying Users to request for confirmation of truthfulness of the encrypted DIDs recorded on the blockchain system. In some embodiments, the methods and devices also support recordation of responses to the requests recorded on the blockchain system. This allows the methods and devices to provide a mechanism for Responders to respond to the recorded requests. In some embodiments, the methods and devices further support a protocol that requires the Responders to provide a proof along with each response. This allows the methods and devices to provide trust for the Relying Users. In this manner, the Relying Users can trust that the Responders are qualified to respond to the recorded requests, thereby eliminating the need for confirmation documents.

[0019] A blockchain system is implemented using a peer-to-peer (P2P) network, in which the nodes communicate directly with each other, e.g., without the need of a fixed, central server. Each node in the P2P network may initiate communication with another node in the P2P network. A blockchain system

maintains one or more blockchains.

[0020] A blockchain is a data structure that stores data, e.g., transactions and the like, in a way that may prevent tampering and manipulation of the data by malicious parties. The transactions stored in this manner may be immutable and subsequently verified. A blockchain includes one or more blocks. Each block is linked to a previous block immediately before it in the blockchain by including a cryptographic hash of the previous block. Each block also may include a timestamp, its own cryptographic hash, and one or more transactions. The transactions, which generally have already been verified by the nodes of the blockchain system, may be hashed and encoded into a data structure, such as a Merkle tree. In a Merkle tree, data at leaf nodes of the tree is hashed, and all hashes in each branch of the tree may be concatenated at a root of the branch. This process continues up the tree to the root of the entire tree, which stores a hash that is representative of all data in the tree. A hash purporting to be of a transaction stored in the tree can be quickly verified by determining whether it is consistent with the structure of the tree.

[0021] A blockchain system includes a network of computing nodes that manage, update, and maintain one or more blockchains. The network may be a public blockchain network, a private blockchain network, or a consortium blockchain network. For example, numerous entities, such as hundreds, thousands, or even millions of entities, can operate in a public blockchain network, and each of the entities operates at least one node in the public blockchain network. Accordingly, the public blockchain network can be considered a public network with respect to the participating entities. Sometimes, a majority of entities (nodes) must sign every block for the block to be valid and added to the blockchain of the blockchain network. Examples of public blockchain networks include particular peer-to-peer payment

networks that leverage a distributed ledger, referred to as blockchain.

[0022] In general, a public blockchain network may support public transactions. A public transaction is shared with all of the nodes in the public blockchain network, and is stored in a global blockchain. A global blockchain is a blockchain replicated across all nodes, and all nodes are in perfect state consensus with respect to the global blockchain. To achieve consensus (e.g., agreement to the addition of a block to a blockchain), a consensus protocol is implemented in the public blockchain network. Examples of consensus protocols include proof-of-work (POW) (e.g., implemented in the some crypto-currency networks), proof-of-stake (POS), and proof-of-authority (POA).

[0023] In general, a private blockchain network may be provided for a particular entity, which centrally controls read and write permissions. The entity controls which nodes are able to participate in the blockchain network. Consequently, private blockchain networks are generally referred to as permissioned networks that place restrictions on who is allowed to participate in the network, and on their level of participation (e.g., only in certain transactions). Various types of access control mechanisms can be used (e.g., existing participants vote on adding new entities, a regulatory authority can control admission).

[0024] In general, a consortium blockchain network may be private among the participating entities. In a consortium blockchain network, the consensus process is controlled by an authorized set of nodes, one or more nodes being operated by a respective entity (e.g., a financial institution, insurance company). For example, a consortium of ten (10) entities (e.g., financial institutions, insurance companies) can operate a consortium blockchain network, each of which operates at least one node in the consortium blockchain network. Accordingly, the consortium blockchain

network can be considered a private network with respect to the participating entities. In some examples, each entity (node) must sign every block in order for the block to be valid, and added to the blockchain. In some examples, at least a sub-set of entities (nodes) (e.g., at least 7 entities) must sign every block in order for the block to be valid, and added to the blockchain.

[0025] FIG. 1 illustrates a schematic diagram of a blockchain system 100, according to an embodiment. Referring to FIG. 1, the blockchain system 100 may include a plurality of nodes, e.g., nodes 102-110, configured to operate on a blockchain 120. The nodes 102-110 may form a network 112, such as a peer-to-peer (P2P) network. Each of the nodes 102-110 may be a computing device, such as a computer or a computer system, configured to store a copy of the blockchain 120, or may be software running on the computing device, such as a process or an application. Each of the nodes 102-110 may have a unique identifier.

[0026] The blockchain 120 may include a growing list of records in the form of data blocks, such as blocks B1-B5 in FIG. 1. Each of the blocks B1-B5 may include a timestamp, a cryptographic hash of a previous block, and data of the present block, which may be transactions such as monetary transactions. For example, as illustrated in FIG. 1, block B5 may include a timestamp, a cryptographic hash of block B4, and transaction data of block B5. Also, for example, a hashing operation may be performed on the previous block to generate the cryptographic hash of the previous block. The hashing operation may convert inputs of various lengths into cryptographic outputs of a fixed length through a hash algorithm, such as SHA-256.

[0027] The nodes 102-110 may be configured to perform an operation on the blockchain 120. For example, when a node, e.g., the node 102, wants to store new data onto the blockchain 120, that node may generate a new block to be added to

the blockchain 120 and broadcast the new block to other nodes, e.g., the nodes 104-110, in the network 112. Based on legitimacy of the new block, e.g., validity of its signature and transactions, the other nodes may determine to accept the new block, such that the node 102 and the other nodes may add the new block to their respective copies of the blockchain 120. As this process repeats, more and more blocks of data may be added to the blockchain 120.

[0028] FIG. 2 illustrates a schematic diagram of a computing device 200 for implementing a node, e.g., the node 102 (FIG. 1), in a blockchain system, according to an embodiment. Referring to FIG. 2, the computing device 200 may include a communication interface 202, a processor 204, and a memory 206.

[0029] The communication interface 202 may facilitate communications between the computing device 200 and devices implementing other nodes, e.g., nodes 104-110 (FIG. 1), in the network. In some embodiments, the communication interface 202 is configured to support one or more communication standards, such as an Internet standard or protocol, an Integrated Services Digital Network (ISDN) standard, etc. In some embodiments, the communication interface 202 may include one or more of a Local Area Network (LAN) card, a cable modem, a satellite modem, a data bus, a cable, a wireless communication channel, a radio-based communication channel, a cellular communication channel, an Internet Protocol (IP) based communication device, or other communication devices for wired and/or wireless communications. In some embodiments, the communication interface 202 may be based on public cloud infrastructure, private cloud infrastructure, hybrid public/private cloud infrastructure.

[0030] The processor 204 may include one or more dedicated processing units, application-specific integrated circuits (ASICs), field-programmable gate arrays

(FPGAs), or various other types of processors or processing units. The processor 204 is coupled with the memory 206 and is configured to execute instructions stored in the memory 206.

[0031] The memory 206 may store processor-executable instructions and data, such as a copy of the blockchain 120 (FIG. 1). The memory 206 may include any type of volatile or non-volatile memory devices, or a combination thereof, such as a static random-access memory (SRAM), an electrically erasable programmable read-only memory (EEPROM), an erasable programmable read-only memory (EPROM), a programmable read-only memory (PROM), a read-only memory (ROM), a magnetic memory, a flash memory, or a magnetic or optical disk. When the instructions in the memory 206 are executed by the processor 204, the computing device 200 may perform an operation on the blockchain 120.

[0032] FIG. 3 illustrates a flow chart of a method 300 for providing decentralized identity verification according to an embodiment. Referring to FIG. 3, multiple users may have accounts on a Blockchain, e.g., the blockchain 120 (FIG. 1). The Blockchain may be implemented to support various types of users, or parties, including, e.g., individuals, businesses, banks, financial institutions, hospitals, as well as other types of companies, organizations, and the like.

[0033] For illustrative purposes, three users are depicted in FIG. 3, which includes a DID Owner, a Relying User, and a Responder. The DID Owner may create a decentralized identifier (DID) and attempt to prove its identity using the DID. The Relying User may receive the DID from the DID Owner, verify the validity of the DID, record a request for confirmation on the Blockchain, and rely on the Blockchain to confirm the truthfulness of the DID. The Responder may notice the request recorded on the Blockchain and provide a response along with a proof on the

Blockchain to help the Relying User confirm the truthfulness of the DID. In some embodiments, the DID Owner, the Relying User, and the Responder may operate according to the method 300.

[0034] At step 302, the DID Owner may generate a DID. In some embodiments, the DID Owner may generate the DID based on various types of information. For example, if the DID Owner is a merchant who sells products on an e-commerce platform, the DID Owner may choose to generate the DID based on the name of the DID Owner and one or more orders the DID Owner had received through the e-commerce platform. In another example, if the DID Owner holds a bank account, the DID Owner may choose to generate the DID based on the name of the DID Owner and an account number associated with the bank account. It is to be understood that the DID Owner may generate different DIDs based on different types of information. However, for illustrative purposes, the following examples may describe a DID Owner who chooses to generate a DID based on information concerning an order the DID Owner received through an e-commerce platform. The DID generated in this manner may contain data fields including, e.g., {name : “ABC,” order_number : “001,” order_year : “2019,” order_month : “September”}.

[0035] In some embodiments, the DID Owner may encrypt the DID at step 302, e.g., using a hash function or an encryption algorithm. Continuing with the example above, the encrypted DID may contain data fields including, e.g., {name : “hsifheuwfh398,” order_number : “eghuefredu09f9,” order_year : “57835fegre4oifnj,” order_month : “8n349t349tnalen”}, where “hsifheuwfh398,” “eghuefredu09f9,” “57835fegre4oifnj,” and “8n349t349tnalen” may represent the encrypted values of “ABC,” “001,” “2019,” and “September,” respectively.

[0036] At step 304, the DID owner may submit the encrypted DID to the

Blockchain for recordation.

[0037] At step 306, the Blockchain may record the encrypted DID submitted by the DID Owner. In some embodiments, the Blockchain may provide the DID Owner with a record number that can be used to retrieve the encrypted DID. In some embodiments, the record number may include an address indicating where the encrypted DID is recorded on the Blockchain. It is to be understood, however, that the record number may be generated in various other manners.

[0038] At step 308, the DID Owner may provide the DID, e.g., {name : “ABC,” order_number : “001,” order_year : “2019,” order_month : “September”}, to the Relying User. The DID Owner may provide the DID to the Relying User for various reasons, including, e.g., to prove the identity of the DID Owner to the Relying User. The Relying User may be, e.g., a service provider or a bank who wants to verify the identity of the DID Owner prior to rendering a service or issuing a loan to the DID Owner.

[0039] At step 310, the Relying User may encrypt the DID received from the DID Owner using the same encryption mechanism the DID Owner used at step 302. The Relying User may then verify the validity of the DID by determining whether the encrypted result as calculated by the Relying User at step 310 matches the encrypted DID recorded on the Blockchain. In some embodiments, the Relying User may retrieve the encrypted DID recorded on the Blockchain using the record number, if provided by the DID Owner, and compare the encrypted DID retrieved from the Blockchain against the encrypted result calculated by the Relying User at step 310. In some embodiments, the Relying User may search whether a record exists on the Blockchain that matches the encrypted result calculated by the Relying User at step 310. In some embodiments, the Relying User may refuse to accept the

identity of the DID Owner if the encrypted result calculated by the Relying User at step 310 does not match any encrypted DIDs recorded on the Blockchain. On the other hand, if the Relying User determines that the encrypted result calculated at step 310 matches the encrypted DID recorded on the Blockchain, the Relying User may proceed to step 312.

[0040] At step 312, the Relying User may submit a request for confirmation to the Blockchain. In some embodiments, the Relying User may request for confirmation of truthfulness of the encrypted DID recorded on the Blockchain. In some embodiments, the Relying User may submit the request with the record number that can be used to retrieve the encrypted DID recorded on the Blockchain. In some embodiments, the Relying User may submit the request with the encrypted DID as its payload, which may include, e.g., {name : “hsifheuwfh398,” order_number : “eghuefredu09f9,” order_year : “57835fegre4oifnj,” order_month : “8n349t349tnalen”}.

[0041] At step 314, the Blockchain may record the request for confirmation. In some embodiments, once the request for confirmation is recorded on the Blockchain, the request may become visible to other users of the Blockchain, including, e.g., a user with data access to the e-commerce platform on which the DID Owner received the order (i.e., the order which was used by the DID Owner to generate the DID at step 302). In some embodiments, the Blockchain may broadcast the request for confirmation on the Blockchain, and users having received the broadcast may retrieve the encrypted DID from the Blockchain and determine whether the encrypted DID contains data fields that they can verify. If a user determines that the encrypted DID contains data fields that the user can verify (e.g., the user has data access to the e-commerce platform depicted in the example above), that user may agree to

serve as the Responder to the request for confirmation and proceed to step 316.

[0042] At step 316, the Responder may retrieve the encrypted DID that needs confirmation from the Blockchain. If the Relying User submitted the request for confirmation with the record number, the Responder may retrieve the encrypted DID using the record number. If the Relying User submitted the request for confirmation with the encrypted DID as its payload, the Responder may retrieve the encrypted DID from the payload.

[0043] At step 318, the Responder may determine the truthfulness of the encrypted DID and generate a response based on the determined truthfulness. In some embodiments, the Responder may determine the truthfulness of the encrypted DID by verifying the encrypted values associated with the data fields contained in the encrypted DID. Continuing with the example above, where the encrypted DID contains, e.g., {name : “hsifheuwfh398,” order_number : “eghuefredu09f9,” order_year : “57835fegre4oifnj,” order_month : “8n349t349tnalen”}, the Responder may determine the truthfulness of this encrypted DID by verifying the encrypted values associated with the name, order number, order year, and order month. Because the Responder has data access to the e-commerce platform on which the DID Owner received the order, the Responder can retrieve the plaintext values of the name (“ABC”), order number (“001”), order year (“2019”), and order month (“September”) from one or more data storage devices utilized by the e-commerce platform. The Responder may then calculate the encrypted values of the name, order number, order year, and order month, respectively, and compare the calculated encrypted values against the encrypted values associated with the data fields contained in the encrypted DID. If the encrypted value calculated for a particular data field matches the encrypted value associated with the corresponding

data field contained in the encrypted DID, then the Responder may determine the truthfulness of that particular data field to be “true.” If the encrypted value calculated for a particular data field does not match the encrypted value associated with the corresponding data field contained in the encrypted DID, then the Responder may determine the truthfulness of that particular data field to be “false.”

[0044] In some embodiments, the Responder may determine the truthfulness of the data fields contained in the encrypted DID and generate a response to indicate the determined truthfulness for each data field individually. Such a response may include, e.g., {name : “true/false,” order_number : “true/false,” order_year : “true/false,” order_month : “true/false”}. In some embodiments, the Responder may determine the truthfulness of the data fields contained in the encrypted DID and indicate the truthfulness of the encrypted DID as a whole. In this manner, if the Responder determines that one of the data fields contained in the encrypted DID is “false,” then the Responder may indicate the truthfulness of the encrypted DID as a whole as “false.” It is to be understood that the Responder may indicate the determined truthfulness in various other manners. For example, in some embodiments, if the Responder does not have the plaintext value for a particular data field contained in the encrypted DID, the Responder may indicate that particular data field as “unverifiable.” It is to be understood that the Responder may also utilize other types of indicators.

[0045] The Responder may also generate a proof at step 318 indicating that the Responder is qualified to determine the truthfulness of the encrypted DID. In some embodiments, the Responder may generate the proof as a zero-knowledge proof. Zero-knowledge proof refers to techniques that allow a prover to prove to a verifier that a statement is true without revealing any information beyond the validity

of the statement itself. At step 318, the Responder is the prover, who may prove to the Blockchain, or a smart contract executing on the Blockchain, which serves as the verifier, that the Responder is qualified to determine the truthfulness of the encrypted DID. The Responder may attempt to prove this by indicating that: (1) the Responder has access to the plaintext values of the data fields contained in the encrypted DID and (2) the Responder is indeed the Responder itself.

[0046] In some embodiments, the Responder and the Blockchain may agree to implement zero-knowledge proof techniques such as Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK) or the like. The Responder may prove to the Blockchain that the Responder knows a secret input w such that for a public input x, a certain relationship between x and w holds true. In some embodiments, the relationship may be defined as an arithmetic circuit. In some embodiments, the arithmetic circuit may be defined based on an equation of polynomials that can be evaluated based on x and w. In some embodiments, a proving key and a verification key may be generated in a setup phase based on the arithmetic circuit and one or more security parameters established for the zero-knowledge proof. One of ordinary skill in the art will understand that the setup phase may be executed by a trusted party, or by multiple independent parties collaboratively using a multi-party computation.

[0047] In some embodiments, the Responder may set w to a value generated based on the plaintext values of the data fields contained in the encrypted DID (to prove that the Responder has access to the plaintext values of these data fields) and the Responder's private key (to prove that Responder is indeed the Responder itself). For example, w may be generated by concatenating the plaintext values of the data fields contained in the encrypted DID and the Responder's private key. In

this manner, the Responder may generate a proof using the secret and public inputs as well as the proving key to prove to the Blockchain that the Responder is in possession of the secret input w. In some embodiments, the Blockchain may be able to verify the proof using the public input and the verification key generated in the setup phase. In some embodiments, if the Blockchain accepts the Responder's proof that the Responder is in possession of the secret input w, the Blockchain may accept the Responder's statements as true.

[0048] At step 320, the Responder may submit the response and the proof to the Blockchain for recordation. In some embodiments, the Blockchain may record the response and the proof submitted by the Responder.

[0049] At step 322, the Blockchain may check the proof to determine whether the Responder is in possession of the secret input w. In some embodiments, the Blockchain may check the proof in response to a verification request submitted by the Responder, and in some embodiments, the Responder may submit the verification request as a transaction separate from the submission of the response and the proof carried out at step 320. In some embodiments, the Blockchain may utilize one or more smart contracts executing on the Blockchain to provide the determination. Smart contracts are computer protocols implemented in the form of computer code that are incorporated into the Blockchain, to facilitate, verify, or enforce the negotiation or performance of contracts. For example, users of the Blockchain may program agreed terms into a smart contract using a programming language, such as C++, Java, Solidity, Python, etc., and when the terms are met, the smart contract may be automatically executed on the Blockchain, e.g., to perform a transaction. Also for example, the smart contract may include a plurality of subroutines or functions, each of which may be a sequence of program instructions

that performs a specific task. The smart contract may be operational code that is fully or partially executed without human interaction.

[0050] In some embodiments, a smart contract may be incorporated into the Blockchain to determine whether the proof is acceptable. The smart contract may verify the proof using the public input and the verification key. If the proof cannot be verified, the smart contract may refuse to accept the response submitted by the Responder as valid. On the other hand, if the proof can be verified, then the smart contract may determine that the proof is acceptable and accept the response submitted by the Responder as valid. In some embodiments, the smart contract may record an indicator (e.g., a flag, a bit, a data field, or the like) indicating the validity of the response. In some embodiments, responses and their validities may become visible to other users of the Blockchain, including, e.g., the Relying User. The Relying User may then proceed to step 324.

[0051] At step 324, the Relying User may retrieve the response from the Blockchain and process the response to determine whether to accept or reject the DID provided by the DID Owner. In some embodiments, the Relying User may process the response only if the response is deemed valid by the smart contract. In some embodiments, the Relying User Responder may determine whether to accept or reject the DID provided by the DID Owner based on the truthfulness indicated in the response. For example, if the response indicates that all data fields contained in the encrypted DID are “true,” the Relying User may conclude that all data fields contained in the DID provided by the DID Owner are “true” and accept the DID provided by the DID Owner. On the other hand, if the response indicates that one or more data fields contained in the encrypted DID is “false,” the Relying User may conclude that one or more data fields contained in the DID provided by the DID

Owner are “false” and refuse to accept the DID provided by the DID Owner. In some embodiments, the Relying User may establish a tolerance threshold, allowing a certain number of data fields to be deemed “false” or “unverifiable” and still be willing to accept the DID provided by the DID Owner. It is to be understood that whether to implement such a tolerance threshold, and how to implement such a tolerance threshold, may vary depending on the Relying User.

[0052] At step 326, the Relying User may inform the DID Owner whether the DID provided by the DID Owner is accepted or rejected.

[0053] It is to be understood that while the examples above depict a DID Owner who chooses to generate a DID based on information concerning an order the DID Owner received through an e-commerce platform, such an implementation is merely provided as an example and is not meant to be limiting. DID Owners may generate DIDs based on various types of information that the DID Owners are willing to use as their identifiers. Furthermore, it is to be understood that the declarations of the functions, variables, and transactions described above are merely presented as examples and are not meant to be limiting.

[0054] It is also to be understood that the method 300 allows the DID Owner to prove its identity to the Relying User by sending its DID to the Relying User in plaintext. In this manner, the method 300 can support DID verification without requiring the Responder to issue any confirmation document, which in turn eliminates the need for the DID Owner to store and manage any confirmation document.

[0055] FIG. 4 illustrates a flow chart of a method 400 for providing decentralized identity verification, according to an embodiment. The method 400 may be performed by one or more nodes in a blockchain system, e.g., the nodes

102-110 in the blockchain system 100 (FIG. 1). The nodes 102-110 in the blockchain system 100 may perform operations on a blockchain, e.g., the blockchain 120 (FIG. 1). The blockchain 120 may be implemented as the Blockchain in the examples described above.

[0056] At step 402, a node, e.g., the node 102, may record an encrypted DID received from a first user. The first user may be, e.g., the DID Owner (FIG. 3). The encrypted DID may include a plurality of encrypted values corresponding to a plurality of data fields. For example, the encrypted DID depicted in the examples above may include {name : “hsifheuwfh398,” order_number : “eghuefredu09f9,” order_year : “57835fegre4oifnj,” order_month : “8n349t349tnalen”}.

[0057] At step 404, the node 102 may record a request for confirmation of truthfulness of the encrypted DID. The request may be received from a second user. The second user may be, e.g., the Relying User (FIG. 3). The Relying User may be, e.g., a service provider or a bank who wants to verify the identity of the DID Owner prior to rendering a service or issuing a loan to the DID Owner. Once the request is recorded, the request may become visible to other users of the blockchain 120, including, e.g., a third user. The third user may be, e.g., the Responder (FIG. 3).

[0058] At step 406, the node 102 may receive a response to the request from the third user along with a proof for proving that the third user is qualified to respond to the request. In some embodiments, the response may include a truthfulness indicator for each of the plurality of data fields included in the encrypted DID. For example, if the encrypted DID includes {name : “hsifheuwfh398,” order_number : “eghuefredu09f9,” order_year : “57835fegre4oifnj,” order_month : “8n349t349tnalen”}, the response may include a truthfulness indicator for each of the plurality of data fields, i.e., name, order_number, order_year, and order_month. In

some embodiments, the truthfulness indicator may have the value of “true” or “false,” indicating whether or not the encrypted value corresponding to each particular data field is truthful. In some embodiments, the truthfulness indicator may also indicate if a particular data field is “unverifiable.” Furthermore, in some embodiments, the response may include a truthfulness indicator for the encrypted DID as a whole. In some embodiments, if the Responder determines that one of the data fields contained in the encrypted DID is “false,” the Responder may indicate the truthfulness of the encrypted DID as a whole as “false.”

[0059] At step 408, the node 102 may record the response. The node 102 may also determine whether the proof is acceptable. In some embodiments, the proof may include a zero-knowledge proof. In some embodiments, the third user may use the proof to prove to the node 102 that the third user is in possession of the secret input described above. In some embodiments, if the node 102 determines that the proof is unacceptable, the node 102 may report an error and/or record an indicator indicating that the response is invalid. On the other hand, if the node 102 determines that the proof is acceptable, the node 102 may record an indicator indicating that the response is valid. The response and its validity indicator may become visible to other users of the blockchain 120, including, e.g., the Relying User. The Relying User may then retrieve and process the response as described above if the response is valid.

[0060] FIG. 5 is a block diagram of a decentralized identity verification apparatus 500, according to an embodiment. The apparatus 500 may be an implementation of a software process, and may correspond to the method 400 (FIG. 4). Referring to FIG. 5, the apparatus 500 may include a receiving module 502, a recording module 504, a determination module 506, and a reporting module 508.

[0061] The receiving module 502 may receive an encrypted DID from a first user. The receiving module 502 may also receive a request from a second user for requesting confirmation of truthfulness of the encrypted DID. The receiving module 502 may provide the encrypted DID and the request to the recording module 504, which may record the encrypted DID and the request in a data storage device, such as a blockchain system.

[0062] The receiving module 502 may further receive from a third user a response to the request and a proof for proving that the third user is qualified to respond to the request. The receiving module 502 may provide the response to the recording module 504, which may record the response. The receiving module 502 may provide the proof to the determining module 504, which may determine whether the proof is acceptable. If the proof is unacceptable, the determining module 506 may request the reporting module 508 to report an error and/or record an indicator indicating that the response is invalid. Otherwise, the determining module 506 may request the recording module 504 to record an indicator indicating that the response is valid. The response and its validity indicator may become visible to the second user, who may then retrieve and process the response as described above if the response is valid.

[0063] Each of the above described modules may be implemented as software, or hardware, or a combination of software and hardware. For example, each of the above described modules may be implemented using a processor executing instructions stored in a memory. Also, for example, each the above described modules may be implemented with one or more application specific integrated circuits (ASICs), digital signal processors (DSPs), digital signal processing devices (DSPDs), programmable logic devices (PLDs), field programmable gate

arrays (FPGAs), controllers, micro-controllers, microprocessors, or other electronic components, for performing the described methods. Further for example, each of the above described modules may be implemented by using a computer chip or an entity, or implemented by using a product having a certain function. In one embodiment, the apparatus 500 may be a computer, and the computer may be a personal computer, a laptop computer, a cellular phone, a camera phone, a smartphone, a personal digital assistant, a media player, a navigation device, an email receiving and sending device, a game console, a tablet computer, a wearable device, or any combination of these devices.

[0064] For an implementation process of functions and roles of each module in the apparatus 500, references can be made to corresponding steps in the above-described methods. Details are omitted here for simplicity.

[0065] In some embodiments, a computer program product may include a non-transitory computer-readable storage medium having computer-readable program instructions thereon for causing a processor to carry out the above-described methods.

[0066] The computer-readable storage medium may be a tangible device that can store instructions for use by an instruction execution device. The computer-readable storage medium may be, for example, but is not limited to, an electronic storage device, a magnetic storage device, an optical storage device, an electromagnetic storage device, a semiconductor storage device, or any suitable combination of the foregoing. A non-exhaustive list of more specific examples of the computer-readable storage medium includes the following: a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM), a static random access

memory (SRAM), a portable compact disc read-only memory (CD-ROM), a digital versatile disk (DVD), a memory stick, a floppy disk, a mechanically encoded device such as punch-cards or raised structures in a groove having instructions recorded thereon, and any suitable combination of the foregoing.

[0067] The computer-readable program instructions for carrying out the above-described methods may be assembler instructions, instruction-set-architecture (ISA) instructions, machine instructions, machine dependent instructions, microcode, firmware instructions, state-setting data, or source code or object code written in any combination of one or more programming languages, including an object oriented programming language, and conventional procedural programming languages. The computer-readable program instructions may execute entirely on a computing device as a stand-alone software package, or partly on a first computing device and partly on a second computing device remote from the first computing device. In the latter scenario, the second, remote computing device may be connected to the first computing device through any type of network, including a local area network (LAN) or a wide area network (WAN).

[0068] The computer-readable program instructions may be provided to a processor of a general-purpose or special-purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the above-described methods

[0069] The flow charts and diagrams in the figures illustrate the architecture, functionality, and operation of possible implementations of devices, methods, and computer program products according to various embodiments of the specification. In this regard, a block in the flow charts or diagrams may represent a software

program, segment, or portion of code, which comprises one or more executable instructions for implementing specific functions. It should also be noted that, in some alternative implementations, the functions noted in the blocks may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the diagrams and/or flow charts, and combinations of blocks in the diagrams and flow charts, may be implemented by special purpose hardware-based systems that perform the specified functions or acts, or combinations of special purpose hardware and computer instructions.

[0070] It is appreciated that certain features of the specification, which are, for clarity, described in the context of separate embodiments, may also be provided in combination in a single embodiment. Conversely, various features of the specification, which are, for brevity, described in the context of a single embodiment, may also be provided separately or in any suitable subcombination or as suitable in any other described embodiment of the specification. Certain features described in the context of various embodiments are not essential features of those embodiments, unless noted as such.

[0071] Although the specification has been described in conjunction with specific embodiments, many alternatives, modifications and variations will be apparent to those skilled in the art. Accordingly, the following claims embrace all such alternatives, modifications and variations that fall within the terms of the claims.

What Is Claimed Is:

1. A computer-implemented method for providing decentralized identity verification, the method comprising:

receiving, from a responder, a response to a request for confirmation of truthfulness of an encrypted decentralized identifier (DID) and a proof for proving the responder is qualified to respond to the request for confirmation, without revealing data used by the responder to generate the response and the proof;

recording the response to the request for confirmation; and

in response to a determination that the proof is acceptable, indicating the response as valid.

2. The method of claim 1, wherein the method is performed by one or more nodes in a blockchain system.

3. The method of any preceding claim, further comprising:

reporting an error in response to a determination that the proof is unacceptable.

4. The method of any preceding claim, wherein the proof is a zero-knowledge proof.

5. The method of any preceding claim, wherein the encrypted DID comprises a plurality of encrypted values corresponding to a plurality of data fields.

6. The method of claim 5, wherein the response to the request for confirmation comprises a truthfulness indicator for each of the plurality of data fields.

7. The method of claim 6, wherein the truthfulness indicator for each particular data field of the plurality of data fields indicates whether or not an encrypted value corresponding to that particular data field is truthful.

8. A device for providing decentralized identity verification, comprising:

one or more processors; and

one or more computer-readable memories coupled to the one or more processors and having instructions stored thereon that are executable by the one or more processors to perform the method of any of claims 1 to 7.

9. An apparatus for providing decentralized identity verification, the apparatus comprising a plurality of modules for performing the method of any of claims 1 to 7.

10. A non-transitory computer-readable medium having stored therein instructions that, when executed by a processor of a device, cause the device to perform the method of any of claims 1 to 7.