

METHODS AND DEVICES FOR MANAGING CONTROL OF ASSET ON BLOCKCHAIN

TECHNICAL FIELD

[0001] The specification relates generally to computer technologies, and more particularly, to methods and devices for managing control of asset on blockchain.

BACKGROUND

[0002] Blockchain systems, also known as distributed ledger systems (DLSS) or consensus systems, may enable participating parties to store data securely and immutably. Blockchain systems may include any DLSSs, 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.

[0003] 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. A blockchain is a data structure for storing data, such as transactions, that may prevent tampering and manipulation of the data by malicious parties.

[0004] Users of a blockchain system may utilize the blockchain system to record

various types of information. For example, users may record information regarding one or more assets owned by or managed by the users. The users may also perform one or more operations on the assets. Furthermore, there may be situations where a user may want to delegate certain responsibilities or authorize another user to perform certain operations on the assets. However, the ability to delegate such responsibilities or to authorize other users to perform such operations may not be readily available in the blockchain system utilized by the user.

SUMMARY

[0005] In one aspect, a computer-implemented method for managing control of asset on blockchain includes receiving, from a user, a request to invoke an operation on an asset recorded on the blockchain and a token issued to the user; obtaining token data and a token signature from the token; determining, based on an issuer of the token, whether the token signature is acceptable; determining, based on the token data, whether the token authorizes the user to invoke the requested operation; determining whether the token is valid; and in response to determinations that the token signature is acceptable, the token authorizes the user to invoke the requested operation, and the token is valid, allowing the user to invoke the requested operation on the asset recorded on the blockchain.

[0006] In another aspect, a device for managing control of asset on blockchain 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: receive, from a user, a request to invoke an operation on an asset recorded on the blockchain and a token issued to the user; obtain token data and a token signature from the token; determine, based on an issuer

of the token, whether the token signature is acceptable; determine, based on the token data, whether the token authorizes the user to invoke the requested operation; determine whether the token is valid; and in response to determinations that the token signature is acceptable, the token authorizes the user to invoke the requested operation, and the token is valid, allow the user to invoke the requested operation on the asset recorded on the blockchain.

[0007] 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 managing control of asset on blockchain. The method includes: receiving, from a user, a request to invoke an operation on an asset recorded on the blockchain and a token issued to the user; obtaining token data and a token signature from the token; determining, based on an issuer of the token, whether the token signature is acceptable; determining, based on the token data, whether the token authorizes the user to invoke the requested operation; determining whether the token is valid; and in response to determinations that the token signature is acceptable, the token authorizes the user to invoke the requested operation, and the token is valid, allowing the user to invoke the requested operation on the asset recorded on the blockchain.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] 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.

[0009] FIG. 1 is a schematic diagram of a blockchain system, according to an embodiment.

[0010] FIG. 2 is a schematic diagram of a computing device for implementing a node in a blockchain system, according to an embodiment.

[0011] FIG. 3 is a flow chart of a method for managing control of asset on blockchain, according to an embodiment.

[0012] FIG. 4 is a flow chart of a method for managing control of asset on blockchain, according to an embodiment.

[0013] FIG. 5 is a block diagram of an apparatus for managing control of asset on blockchain, according to an embodiment.

DETAILED DESCRIPTION

[0014] Embodiments of the specification provide methods and devices for managing control of asset on blockchain. The methods and devices may utilize tokens issued off-chain to facilitate management of assets recorded on a blockchain. The methods and devices may allow users requesting to invoke operations on assets recorded on the blockchain to use the tokens to prove that the users are authorized to invoke such operations. The methods and devices may utilize one or more smart contracts executing on the blockchain to verify the validity of the tokens and verify whether the tokens authorize the users to perform the requested operations. The methods and devices may utilize such tokens and smart contracts to efficiently manage control of assets recorded on the blockchain.

[0015] Embodiments disclosed in the specification have one or more technical effects. In some embodiments, the methods and devices utilize tokens issued off-chain to facilitate management of assets recorded on a blockchain. This allows the

methods and devices to reduce the need to maintain a large authorization table on the blockchain. This also allows the methods and devices to eliminate the need to process authorization transactions, thereby improving the effective throughput of the authorization process. In some embodiments, the methods and devices allow users requesting to invoke operations on assets recorded on the blockchain to use the tokens to prove that the users are authorized to invoke such operations. This allows the methods and devices to verify the tokens without the need to search through an authorization table, further improving the efficiency of the authorization process. In some embodiments, the methods and devices may utilize one or more smart contracts executing on the blockchain to verify the validity of the tokens and verify whether the tokens authorize the users to perform the requested operations. This further eliminates the need for the methods and devices to search through an authorization table, improving the efficiency of the authorization process.

[0016] A blockchain is a data structure that stores data, e.g., transactions, 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.

[0017] 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.

[0018] 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).

[0019] 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).

[0020] 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.

[0021] 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.

[0022] 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.

[0023] 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.

[0024] 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.

[0025] 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.

[0026] 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.

[0027] 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.

[0028] Users of a blockchain system, e.g., the blockchain system 100, may utilize the blockchain system 100 to record various types of information. For example,

in some embodiments, users may utilize the blockchain system 100 to record information regarding assets owned by or managed by the users. The blockchain system 100 may support recordation of different types of assets, including, e.g., monetary assets, personal properties, real properties, intellectual properties, contracts, and the like. In some embodiments, each asset recorded on the blockchain system 100 may be associated with a unique identifier.

[0029] In some embodiments, the assets recorded on the blockchain system 100 may be assigned one or more managers, who may be responsible to manage the assets recorded on the blockchain system 100. In some embodiments, the assigned managers may include the owners of the assets. In some embodiments, a manager may be assigned to manage one or more types of assets that may or may not be owned by the manager. For example, a company may own different types of assets, including, e.g., desktops, laptops, printers, stamps, desks, chairs, and the like. The company may record these assets on the blockchain system 100 and assign an IT manager to manage different types of assets, including, e.g., the desktops, laptops, and printers. The company may assign a different manager, e.g., a facilities manager, to manage the desks and chairs, and assign yet another manager, e.g., a secretary, to manage the stamps. In another example, a casino may own different types of assets, including, e.g., slot machines, gaming tablets, electronic gambling machines, chips, and the like. The casino may record these assets on the blockchain system 100 and assign different managers to manage the different types of assets recorded.

[0030] For illustrative purposes, the relationship between a set of sample assets and their corresponding managers are depicted below. In this example, five types of assets, identified by asset types 1, 2, 3, 4, and 5, are recorded on the blockchain system 100.

Asset Type	1	2	3	4	5
Manager	ID_1	ID_2	ID_3	ID_4	ID_5
Public Key	PK_1	PK_2	PK_3	PK_4	PK_5

[0031] As shown in the table above, each of the five types of assets may be managed by a manager who has an account on the blockchain system 100. For illustrative purposes, these managers are identified as ID_i , and their public keys are identified as PK_i . As will be described below, each manager ID_i may utilize its corresponding public key PK_i to generate a signature to facilitate authorization of various operations.

[0032] Also, for illustrative purposes, an asset recorded on the blockchain system 100 may be depicted as follows, where the Asset ID may uniquely identify the asset, the Asset Type may identify the type of the asset, and the Other Data Fields may record optional information about the asset, including, e.g., location, serial number, manufacture, owner, and the like.

Asset ID	Asset Type	Other Data Fields
001	2	...

[0033] In some embodiments, the managers may perform various types of operations on the assets recorded on the blockchain system 100, including, e.g., acquire, receive, move (location change), transfer (ownership change), repair, return, discard, destroy, and the like. For example, if a desktop managed by an IT manager has been moved from one location to another, the IT manager may invoke a move operation to update the location of that desktop asset recorded on the blockchain system 100. For security reasons, in some embodiments, only the manager of an

asset may be allowed to invoke certain operations on that asset. However, there may be situations where the manager may want to (or need to) delegate certain responsibilities to one or more authorized users, granting such authorized users the abilities to invoke certain operations on one or more assets on behalf of the manager. For example, in some instances, the manager may want to grant a batch authorization that gives an authorized user the ability to invoke certain operations on all assets of a particular type. In another example, the manager may want to grant a single authorization that gives an authorized user the ability to invoke certain operations on a specific asset identified by the manager. Furthermore, in some instances, the manager may want to grant a single-use authorization so that the authorized user can only invoke the authorized operation(s) once. Alternatively, the manager may want to grant a multi-use authorization so that the authorized user can invoke the authorized operation(s) until the authorization expires upon a particular termination condition.

[0034] In some embodiments, the blockchain system 100 may support the granting of the various authorizations by recording, on the blockchain system 100, an authorization table. For example, the blockchain system 100 may utilize the authorization table to record all authorizations granted by all managers. In some embodiments, each record in the authorization table may include information containing: (1) the identification(s) of the authorized user(s), (2) the asset type associated with the authorization, (3) the asset identifier associated with the authorization (if the authorization is a batch authorization, the asset identifier may be set to NULL), (4) the validity of the authorization (indicating whether the authorization is valid only once or expires upon a particular termination condition), and (5) one or more operations the authorized user is authorized to invoke. In this manner, a manager who wants to grant another user a batch or a single authorization may submit

an authorization transaction to the blockchain system 100. Upon receiving the authorization transaction, the blockchain system 100 may add an appropriate entry to the authorization table, which can be utilized by the blockchain system 100 to verify whether a particular user is authorized by the manager to invoke a particular operation.

[0035] In some embodiments, the blockchain system 100 may support another method for managing control of assets on the blockchain system 100. FIG. 3 illustrates a flow chart of such a method 300, according to an embodiment. As will be described below, the method 300 may reduce the need to maintain an authorization table on the blockchain system 100, thereby reducing the cost of storage. The method 300 may also eliminate the need for the managers to submit authorization transactions to the blockchain system 100, thereby improving the effective throughput of the authorization process. The method 300 may further eliminate the need to search through the authorization table to determine whether a particular user is authorized to invoke a particular operation, further improving the efficiency of the authorization process.

[0036] For illustrative purposes, a Blockchain, e.g., the blockchain 120 (FIG. 1), is depicted in FIG. 3. The Blockchain may be implemented to support various types of users, or parties, including, e.g., companies, businesses, banks, financial institutions, individuals, employees, as well as other types of organizations, and the like. For illustrative purposes, two users, referred to as a Manager and an Authorized User, are depicted.

[0037] At step 302, the Manager may initiate an authorization process. In some embodiments, the Manager may initiate the authorization process by generating and issuing an off-chain authorization token to the Authorized User. In some embodiments, the token may include token data *data* and a token signature σ . In some embodiments, the token signature σ may include a signature generated based

on the token data *data* and the Manager's public key $PK_{manager}$, e.g., $\sigma = \text{SIGN}(PK_{manager}, data)$. In some embodiments, the token data *data* may include the following data fields:

Data Fields	Description
tokenID	Token Identifier
tokenOwner	Identifier of the token owner, e.g., Authorized User
assetType	Specifies the type of asset the Authorized User is authorized to operate on
assetID	Specifies the asset the Authorized User is authorized to operate on
operations	Specifies the functions, methods, or operations the Authorized User is authorized to invoke
validity	Indicates whether the token is valid only once or expires upon a particular termination condition

[0038] In some embodiments, the tokenID may include an identifier (e.g., a universally unique identifier, or UUID) that can be used to uniquely identify the token. Similarly, the tokenOwner may include an identifier that can be used to uniquely identify the Blockchain account associated with the Authorized User. The assetType may include any type of data (e.g., strings, integers, alphanumericals, enumerated types, and the like) that can be used to identify the type(s) of assets the Authorized User is authorized to operate on. Similarly, the assetID may include any type of data that can be used to identify the asset(s) the Authorized User is authorized to operate on. In some embodiments, the Manager may set the assetID to identify the specific asset(s) the Authorized User is authorized to operate on. Alternatively, the Manager

may set the assetID to NULL (or another predetermined value) to indicate that the Authorized User is authorized to operate on all assets of the type(s) specified by the assetType.

[0039] In some embodiments, the Manager may specify one or more operations the Authorized User is authorized to invoke. The Manager may also use the validity field to specify whether the token is valid for just a single use, or whether the token is set up to expire upon a particular termination condition. For example, the Manager may set the validity to 0 (or another predetermined value) to indicate that the token is valid for just a single use. Alternatively, the Manager may use the validity field to specify a termination condition (e.g., a timestamp), allowing the token to be recognized as invalid automatically upon the occurrence of the termination condition. In still another example, the Manager may set the validity to a specific value to indicate that the token is non-expiring. It is to be understood that the Manager may set the validity of the token in various other manners not explicitly described in the examples above.

[0040] For illustrative purposes, suppose that the Manager wants to grant a single use batch authorization that gives the Authorized User the ability to perform one of the operations OP1, OP2, or OP3 on any asset of asset type “2” once. The Manager may generate and issue an off-chain authorization token to the Authorized User with token data *data* specifying the following:

Data Fields	Data Content
tokenID	token01
tokenOwner	Authorized User’s ID
assetType	2
assetID	NULL
operations	OP1, OP2, OP3

validity	0
----------	---

[0041] The Manager may also generate a token signature σ by signing the token data $data$ using the Manager's public key $PK_{manager}$, i.e., $\sigma = \text{SIGN}(PK_{manager}, data)$. The Manager may then provide the token $(data, \sigma)$ containing the token data $data$ and the token signature σ to the Authorized User through an off-chain communication channel, which may include various types of wired or wireless communication channels, such as the Internet, electronic mails, file transfer applications, text messages, and the like.

[0042] At step 304, the Authorized User may send a request to invoke one of the operations (e.g., operation OP1) on an asset of asset type "2." In some embodiments, the Authorized User may provide/specify, in the request, parameters including an identifier of a user sending the request (e.g., the Authorized User), and the asset type and the asset identifier for invoking the operation. In some embodiments, the Blockchain may also require the Authorized User to specify/provide the token $(data, \sigma)$ issued by the Manager to the Blockchain to prove that the Authorized User is authorized to invoke the operation.

[0043] At step 306, upon receiving the Authorized User's request to invoke the operation on the asset, the Blockchain may verify whether the token $(data, \sigma)$ provided by the Authorized User is valid and whether it authorizes the Authorized User to invoke the requested operation on the asset. In some embodiments, the Blockchain may utilize one or more smart contracts executing on the Blockchain to perform the verification. 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 agreed terms or conditions. 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.

[0044] In some embodiments, the Blockchain, through one or more smart contracts executing on the Blockchain, may receive as input the token $(data, \sigma)$, the identification (e.g., the name) of the operation the Authorized User is requesting to invoke, and one or more parameters the Authorized User provided/specified in its request to invoke the operation. In some embodiments, the parameters may include the asset type and the asset identifier. In this manner, the Blockchain, through one or more smart contracts executing on the Blockchain, may execute a token access control method, described below with reference to steps 308 through 316, to determine whether the token $(data, \sigma)$ is valid and whether the token authorizes the Authorized User to invoke the requested operation (e.g., OP1 in the example above).

[0045] At step 308, the Blockchain may parse the token $(data, \sigma)$ to obtain the token data $data$ (including, e.g., tokenID, tokenOwner, assetType, assetID, operations, and validity, as described above) and the token signature σ . At step 310, the Blockchain may identify the issuer of the token, i.e., the Manager assigned to manage the assetType, and retrieve the public key $PK_{manager}$ associated with the Manager. At step 312, the Blockchain may verify whether the token signature σ is valid based on the token data $data$ and the Manager's public key $PK_{manager}$. For example, the Blockchain may sign the token data $data$ using the Manager's public key

$PK_{manager}$ and compare the signed token data $SIGN(PK_{manager}, data)$ to the token signature σ . If $\sigma \neq SIGN(PK_{manager}, data)$, the Blockchain may refuse to proceed further because the signature σ associated with the token $(data, \sigma)$ may have been forged or the token data $data$ may have been modified after it was signed by the Manager. Otherwise, if $\sigma = SIGN(PK_{manager}, data)$, the Blockchain may accept the token signature σ as valid.

[0046] At step 314, the Blockchain may determine whether the token authorizes the Authorized User to invoke the requested operation. In some embodiments, the Blockchain may make the determination at least partially based on whether the identifier of the tokenOwner specified in the token matches the identifier of the user who is requesting to invoke the operation. If the tokenOwner specified in the token does not match the user who is requesting to invoke the operation, the Blockchain may refuse to proceed further because the mismatch indicates that the user is requesting to use a token issued to someone else. In this manner, the method 300 can prevent unauthorized users from stealing tokens and performing unauthorized operations. On the other hand, if the identifier of the tokenOwner specified in the token matches the identifier of the user who is requesting to invoke the operation, e.g., the Authorized User, the Blockchain may allow the method 300 to proceed.

[0047] In some embodiments, the Blockchain may determine whether the token authorizes the Authorized User to invoke the requested operation at least partially based on whether the assetType specified in the token matches the asset type the Authorized User is attempting to operate on. If the assetType specified in the token does not match the asset type the Authorized User is attempting to operate on (e.g., if the assetType specified in the token is “2” and the Authorized User is attempting to invoke the operation to operate on an asset having a different asset type), the

Blockchain may refuse to proceed further.

[0048] In some embodiments, the Blockchain may determine whether the token authorizes the Authorized User to invoke the requested operation at least partially based on whether the assetID specified in the token matches the identifier of the asset the Authorized User is attempting to operate on. In some embodiments, the Blockchain may recognize the token to be a batch authorization token if the token does not specify an assetID, in which case the Blockchain may not need to determine whether the assetID specified in the token matches the identifier of the asset the Authorized User is attempting to operate on. However, if the token specifies an assetID, and the specified assetID does not match the identifier of the asset the Authorized User is attempting to operate on (e.g., if the assetID specified in the token is “001” and the Authorized User is attempting to invoke the operation to operate on an asset having a different asset identifier), the Blockchain may refuse to proceed further.

[0049] In some embodiments, the Blockchain may determine whether the token authorizes the Authorized User to invoke the requested operation, e.g., operation OP1, at least partially based on whether the requested operation OP1 is one of the operations specified in the token. If the requested operation OP1 is not one of the operations specified in the token, the Blockchain may refuse to proceed further.

[0050] In some embodiments, the Blockchain may determine that the token authorizes the Authorized User to invoke the requested operation if (1) the tokenOwner specified in the token matches the user who is requesting to invoke the operation, (2) the assetType specified in the token matches the asset type the user is attempting to operate on, (3) the token does not specify an assetID, or the token specifies an assetID and the specified assetID matches the identifier of the asset the

user is attempting to operate on, and (4) the requested operation is one of the operations specified in the token.

[0051] At step 316, the Blockchain may determine whether the token is valid. In some embodiments, if the token is issued as a single use token, the Blockchain may check whether the token has already been spent. If not, the Blockchain may determine that the token is valid, but should be recognized as “spent” once the Authorized User performs the requested operation. In some embodiments, the Blockchain may add the token to a spent token list or a spent token hash map so that the token can be recognized as “spent” once the Authorized User performs the requested operation. On the other hand, if the token has already been spent (e.g., the token is already in the spent token list), the Blockchain may determine that the token is invalid and refuse to allow the Authorized User to perform the requested operation. In some embodiments, if the token is configured to expire upon a particular termination condition, the Blockchain may determine whether the termination condition has occurred, and if so, the Blockchain may determine that the token is invalid and refuse to allow the Authorized User to perform the requested operation.

[0052] In some embodiments, the token access control method described above with reference to steps 308 through 316 may be implemented in accordance with the following pseudocode. It is to be understood, however, that the pseudocode illustrated below is merely provided as an example and is not meant to be limiting.

```
token_access_control(token, currentOperation, currentAssetType, currentAssetID) {  
    (tokenId, tokenOwner, assetType, assetID, operations, validity) =  
    parse_token_data(token);  
    get manager's public key based on the assetType;  
    VerifySignature(token_data, token_signature, manager's public key);  
    Abort if tokenOwner != the blockchain account id of the caller;  
    Abort if currentAssetType != assetType;  
    Abort if assetID != "" && assetID != currentAssetID;  
    Abort if operations does not include currentOperation being invoked;  
    If validity = 0 {
```

```

        Abort if tokenID has already been spent;
        Mark tokenID as spent;
    } else {
        Abort if a termination condition has occurred;
    }
}

```

[0053] In some embodiments, if the token_access_control method can be executed successfully (i.e., not aborted), the Blockchain may, at step 318, allow the Authorized User to perform the requested operation on the asset specified. Also, in some embodiments, if the token is a single use token, the Blockchain may add the token to a spent token list or a spent token hash map so that the token can be recognized as “spent” once the Authorized User performs the requested operation.

[0054] As described above, the method 300 may reduce the need to maintain an authorization table on the Blockchain, thereby reducing the cost of storage. The method 300 may also eliminate the need for the managers to submit authorization transactions to the Blockchain. Instead, the authorizations can be issued using off-chain tokens. The method 300 may further eliminate the need to search through an authorization table to determine whether a particular user is authorized to invoke a particular operation. Instead, the Blockchain may use one or more smart contracts to check whether a token is valid and whether it authorizes a user to invoke the requested operation, thereby improving the efficiency of the authorization process.

[0055] FIG. 4 illustrates a flow chart of a method 400 for managing control of asset on blockchain, 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 receive a request to invoke an operation on an asset recorded on the blockchain. The node 102 may receive the request from a user, e.g., the Authorized User depicted in FIG. 3. The node 102 may also receive a token issued to the user. As described above, the token may be issued to the user by an issuer, e.g., the Manager depicted in FIG. 3.

[0057] At step 404, the node 102 may obtain token data and a token signature from the token. In some embodiments, the node 102 may obtain the token data and the token signature by parsing the token. In some embodiments, the obtained token data may include a token identifier, an identifier of a token owner, an asset type, an asset identifier, one or more operations the token owner is authorized to invoke, and a validity value indicating whether the token is valid only once or expires upon a particular termination condition.

[0058] At step 406, the node 102 may identify the issuer of the token. In some embodiments, the node 102 may identify the issuer based on the issuer's association with the asset type specified in the token data. For example, in some embodiments, the node 102 may identify the manager assigned to manage the asset type specified in the token data as the issuer of the token.

[0059] At step 408, the node 102 may determine whether the token signature is acceptable. In some embodiments, the node 102 may retrieve the public key associated with the issuer and sign the token data using the issuer's public key. If the signed token data matches the token signature obtained from the token, the node 102 may accept the token signature as valid. Otherwise, the node 102 may refuse to accept the token signature and refuse to proceed further.

[0060] At step 410, the node 102 may determine whether the token authorizes the user to invoke the requested operation. In some embodiments, the user

requesting to invoke the operation may be associated with an identifier (e.g., the user's account identifier), and the node 102 may determine whether the token authorizes the user to invoke the requested operation by determining whether the token owner specified in the token data matches the identity of the user requesting to invoke the operation. If a mismatch is detected, the node 102 may refuse to proceed further. In some embodiments, the node 102 may determine whether the token authorizes the user to invoke the requested operation by determining whether the asset type specified in the token data matches the asset type the user is attempting to operate on. If a mismatch is detected, the node 102 may refuse to proceed further. In some embodiments, the node 102 may determine whether the token authorizes the user to invoke the requested operation by determining whether the asset identifier specified in the token data matches the asset identifier the user is attempting to operate on. If a mismatch is detected, the node 102 may refuse to proceed further. In some embodiments, the node 102 may determine whether the token authorizes the user to invoke the requested operation by determining whether the requested operation is one of the operation(s) specified in the token data. If the requested operation is not one of the operation(s) specified in the token data, the node 102 may refuse to proceed further.

[0061] In some embodiments, the node 102 may determine that the token authorizes the user to invoke the requested operation when (1) the token owner specified in the token data matches the user requesting to invoke the operation, (2) the asset type specified in the token data matches the asset type the user is attempting to operate on, (3) the token does not specify an asset identifier, or the asset identifier specified in the token data matches the asset identifier the user is attempting to operate on, and (4) the requested operation is one of the operation(s) specified in the

token data.

[0062] At step 412, the node 102 may determine whether the token is valid. In some embodiments, if the token is issued as a single use token, the node 102 may determine whether the token is valid by determining whether the token has been spent already. If not, the node 102 may determine that the token is valid, but should be recognized as “spent” once the user performs the requested operation. In some embodiments, the node 102 may add the token to a spent token list or a spent token hash map so that the token can be recognized as “spent” once the user performs the requested operation. On the other hand, if the token has already been spent (e.g., the token is already in the spent token list), the node 102 may determine that the token is invalid and refuse to allow the user to perform the requested operation. In some embodiments, if the token is configured to expire upon a particular termination condition, the node 102 may determine whether the token is valid by determining whether a termination condition associated with the token has occurred. If so, the node 102 may determine that the token is invalid and refuse to allow the user to perform the requested operation.

[0063] At step 414, in response to determinations that the token signature is acceptable, the token authorizes the user to invoke the requested operation, and the token is valid, the node 102 may allow the user to invoke the requested operation on the asset recorded on the blockchain.

[0064] FIG. 5 is a block diagram of an apparatus 500 for managing control of asset on blockchain, 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 processing module 504, and a determination module 506.

[0065] The receiving module 502 may receive, from a user, a request to invoke an operation on an asset recorded on the blockchain and a token issued to the user. The receiving module 502 may provide the received request and the token to the processing module 504, and the processing module 504 may process the token to obtain token data and a token signature from the token. In some embodiments, the processing module 504 may also identify an issuer of the token based on token data obtained by the processing module 504.

[0066] The determination module 506 may then determine whether the token signature is acceptable, whether the token authorizes the user to invoke the requested operation, and whether the token is valid, as described above. In response to determinations that the token signature is acceptable, the token authorizes the user to invoke the requested operation, and the token is valid, the determination module 506 may notify the processing module 504, which may allow the user to invoke the requested operation on the asset recorded on the blockchain.

[0067] 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.

[0068] 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.

[0069] 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.

[0070] 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.

[0071] 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).

[0072] 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.

[0073] 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.

[0074] 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.

[0075] 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 managing control of asset on blockchain, the method comprising:

receiving, from a user, a request to invoke an operation on an asset recorded on the blockchain and a token issued to the user;

obtaining token data and a token signature from the token, wherein the token data comprises one or more operations a token owner is authorized to invoke;

determining, based on an issuer of the token, whether the token signature is acceptable;

determining, based on the token data, whether the token authorizes the user to invoke the requested operation by determining whether the requested operation is one of the one or more operations specified in the token data;

determining whether the token is valid; and

in response to determinations that the token signature is acceptable, the token authorizes the user to invoke the requested operation, and the token is valid, allowing the user to invoke the requested operation on the asset recorded on the blockchain.

2. The method of claim 1, wherein the token data comprises: a token identifier, an identifier of the token owner, an asset type, an asset identifier, and a validity of the token.

3. The method of claim 2, further comprising:

identifying the issuer of the token based on the issuer's association with the asset type specified in the token data.

4. The method of any of claims 2 and 3, wherein the user requesting to invoke the operation is associated with an identifier, and determining whether the token authorizes the user to invoke the requested operation further comprises:

determining that the identifier of the token owner specified in the token data matches the identifier of the user requesting to invoke the operation.

5. The method of any of claims 2 through 4, wherein the received request includes an asset type the user is attempting to operate on, and determining whether the token authorizes the user to invoke the requested operation further comprises:

determining that the asset type specified in the token data matches the asset type the user is attempting to operate on.

6. The method of any of claims 2 through 5, wherein the received request includes an asset identifier the user is attempting to operate on, and determining whether the token authorizes the user to invoke the requested operation further comprises:

determining that the asset identifier specified in the token data matches the asset identifier the user is attempting to operate on.

7. The method of any of claims 2 through 6, wherein determining whether the token is valid further comprises:

determining that the token has not been spent already.

8. The method of any of claims 2 through 6, wherein determining whether the token is valid further comprises:

determining that a termination condition associated with the token has not occurred.

9. A device for managing control of asset on blockchain, 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 8.

10. An apparatus for managing control of asset on blockchain, the apparatus comprising a plurality of modules for performing the method of any of claims 1 to 8.

11. 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 8.