

METHODS AND DEVICES FOR PROVIDING PRIVACY-PRESERVING PROOF OF ACCOUNTS RECEIVABLE

TECHNICAL FIELD

[0001] The specification relates generally to computer technologies, and more particularly, to methods and devices for providing privacy-preserving proof of accounts receivable.

BACKGROUND

[0002] Accounts receivable refers to the balance due to a party for goods or services delivered but not yet paid for by its customers. There are many situations where a seller may want to prove that it has enough accounts receivable (e.g., greater than a certain threshold). For example, the seller may want to prove to a buyer that the seller has enough accounts receivable (e.g., payments pending from other orders) so that the buyer can gauge the trustworthiness of the sellers. The buyer may deem the seller to be trustworthy if the seller has enough accounts receivable because such sellers are less likely to be out of business, and it also tends to show that the seller has more business establishments with different buyers and possibly higher reputation in the industry. In another example, the seller may want to apply for financing (e.g., obtain loans or investments from banks, financial institutions, or investors), and the seller may want to prove to the lender that it has enough accounts receivable.

[0003] 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 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.

[0004] 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.

[0005] Users of a blockchain system may utilize the blockchain system to record various types of information. For example, a seller may use the blockchain system to record invoices issued to buyers. A buyer may submit one or more purchase orders to the blockchain system, and the purchase orders may be accepted by the seller to show that the seller has accounts receivable pending from its purchase orders. However, because data recorded on the blockchain system is publicly visible, sensitive information such as the identities of buyers and sellers and the amount listed in the invoices or the purchase orders may be exposed to the public.

SUMMARY

[0006] In one aspect, a computer-implemented method for providing privacy-preserving proof of accounts receivable includes: receiving a plurality of encrypted data entries from a plurality of users; recording the plurality of encrypted data entries

on a blockchain; receiving an anonymity data set from a particular user of the plurality of users, the anonymity data set comprising at least one encrypted data entry associated with the particular user and at least one encrypted data entry associated with another user of the plurality of users; receiving a commitment of a sum of an amount specified in the at least one encrypted data entry associated with the particular user; and recording the anonymity data set and the commitment on the blockchain to facilitate a verification of a range proof generated by the particular user for proving that the sum is greater than or equal to a threshold.

[0007] In another aspect, a device for providing privacy-preserving proof of accounts receivable 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 a plurality of encrypted data entries from a plurality of users; record the plurality of encrypted data entries on a blockchain; receive an anonymity data set from a particular user of the plurality of users, the anonymity data set comprising at least one encrypted data entry associated with the particular user and at least one encrypted data entry associated with another user of the plurality of users; receive a commitment of a sum of an amount specified in the at least one encrypted data entry associated with the particular user; and record the anonymity data set and the commitment on the blockchain to facilitate a verification of a range proof generated by the particular user for proving that the sum is greater than or equal to a threshold.

[0008] 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 privacy-preserving proof of accounts receivable. The method includes: receiving a plurality of encrypted data entries from

a plurality of users; recording the plurality of encrypted data entries on a blockchain; receiving an anonymity data set from a particular user of the plurality of users, the anonymity data set comprising at least one encrypted data entry associated with the particular user and at least one encrypted data entry associated with another user of the plurality of users; receiving a commitment of a sum of an amount specified in the at least one encrypted data entry associated with the particular user; and recording the anonymity data set and the commitment on the blockchain to facilitate a verification of a range proof generated by the particular user for proving that the sum is greater than or equal to a threshold.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] 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.

[0010] FIG. 1 is a schematic diagram of a blockchain system, according to an embodiment.

[0011] FIG. 2 is a schematic diagram of a computing device for implementing a node in a blockchain system, according to an embodiment.

[0012] FIG. 3 is a flow chart of a method for providing privacy-preserving proof of accounts receivable, according to an embodiment.

[0013] FIG. 4 is a flow chart of a method for providing privacy-preserving proof of accounts receivable, according to an embodiment.

[0014] FIG. 5 is a block diagram of an apparatus for providing privacy-preserving proof of accounts receivable, according to an embodiment.

DETAILED DESCRIPTION

[0015] Embodiments of the specification provide methods and devices for providing privacy-preserving proof of accounts receivable. The methods and devices may utilize blockchain systems to record information needed to facilitate proof of accounts receivable. The methods and devices may allow the recorded information to be protected using encryption or commitment schemes. The methods and devices may provide operations to be performed by various users (including, e.g., sellers, buyers, verifier, etc.) to facilitate proof of accounts receivable. The methods and devices may also support use of anonymity data sets, which may allow the sellers to prove that their accounts receivable satisfy certain conditions without revealing any sensitive information. Furthermore, the methods and devices may utilize updatable public keys to ensure that information recorded on blockchain systems cannot be linked together.

[0016] Embodiments disclosed in the specification have one or more technical effects. In some embodiments, the methods and devices allow users to record information needed to facilitate privacy-preserving proof of accounts receivable on a blockchain system. This allows the methods and devices to record the information in a data structure that can prevent tampering and manipulation by malicious parties. In some embodiments, the methods and devices allow the recorded information to be protected using encryption or commitment schemes. This allows the methods and devices to preserve user privacy because the users do not need to record sensitive financial information in plaintext on the blockchain system. In some embodiments, the methods and devices can support operations and validations based on the commitment values recorded on the blockchain system. This allows the methods and

devices to provide a mechanism for the users to prove their accounts receivable values without exposing the underlying financial information on the blockchain system. In some embodiments, the methods and devices may support use of anonymity data sets, which may allow the sellers to prove that their accounts receivable satisfy certain conditions without revealing any sensitive information, including, e.g., the number of orders associated with a seller, the identities of such orders, the amount of each order, the total amount of all the orders associated with the seller, and the like. Furthermore, in some embodiments, the methods and devices may utilize updatable public keys to ensure that information recorded on blockchain systems cannot be linked together by third parties, thereby further preserving user privacy.

[0017] 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.

[0018] 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.

[0019] 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).

[0020] 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).

[0021] 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.

[0022] 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.

[0023] 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.

[0024] 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.

[0025] 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.

[0026] 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.

[0027] 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.

[0028] 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.

[0029] 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, one or more users may utilize the blockchain system 100 to record information that can be used to facilitate privacy-preserving proof of accounts receivable. For example, a seller may use the blockchain system 100 to record

invoices issued to buyers. Buyers may also submit purchase orders to the blockchain system 100 for recordation. The recorded information may be utilized to facilitate privacy-preserving proof of accounts receivable for the seller.

[0030] FIG. 3 illustrates a flow chart of a method 300 for providing privacy-preserving proof of accounts receivable, 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., buyers, sellers, suppliers, banks, financial institutions, businesses, individuals, as well as other types of companies, organizations, and the like.

[0031] For illustrative purposes, multiple users are depicted in FIG. 3, including a Buyer, a Seller, a Trusted Third Party, and a Verifier. At step 302, the Buyer may generate an updatable public key (UPK), referred to as the Buyer's UPK. At step 304, the Seller may generate an UPK, referred to as the Seller's UPK, in similar manners. As will be described below, in some embodiments, the public keys of the Buyer and the Seller may be updated for every new order so that a third party may not link different orders together, thereby protecting the privacy of the Buyer and the Seller.

[0032] In some embodiments, the Buyer and the Seller may each generate a public key pk and a private key sk , and use a randomness r and the private key sk to update the public key pk based on an updatable public key system disclosed in Fauzi et al., "Quisquis: A New Design for Anonymous Cryptocurrencies," Advances in Cryptology—ASIACRYPT 2019, 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part I (pp. 649-678), which is herein incorporated by reference in its entirety. As disclosed in Fauzi et al., an updatable public key system may be

defined to include the following functions, where:

$(pk, sk) \xleftarrow{\$} \text{Gen}(1^\kappa)$ takes as input a security parameter κ and outputs a public

key $pk \in \mathcal{PK}$ and a secret key $sk \in \mathcal{SK}$;

$pk' \xleftarrow{\$} \text{Update}(pk)$ takes as input $pk \in \mathcal{PK}$ and $sk \in \mathcal{SK}$ and outputs a new public key pk' ;

$0/1 \leftarrow \text{VerifyKP}(pk, sk)$ takes as input $pk \in \mathcal{PK}$ and $sk \in \mathcal{SK}$ and checks whether or not (pk, sk) is a valid key pair; and

$0/1 \leftarrow \text{VerifyUpdate}(pk', pk, r)$ takes as input public keys pk', pk and randomness r and checks if pk' was output by $\text{Update}(pk; r)$.

[0033] At step 306, the Buyer may place a purchase order with the Seller and generate an encrypted purchase order, and at step 308, the Buyer may submit the encrypted purchase order to the Blockchain for recordation. For illustrative purposes, suppose that the Buyer's UPK is $\overline{pk} = (\bar{g}, \bar{h})$ and the Seller's UPK is $\widetilde{pk} = (\tilde{g}, \tilde{h})$, in some embodiments, before the Buyer submits the encrypted purchase order to the Blockchain, the Buyer may update both the Buyer's public key and the Seller's public key, and encrypt the amount a specified in the purchase order as follows:

$$\overline{pk}' = \text{Update}(\overline{pk}; \bar{r}) = (\bar{g}^{\bar{r}}, \bar{h}^{\bar{r}}),$$

$$\widetilde{pk}' = \text{Update}(\widetilde{pk}; \tilde{r}) = (\tilde{g}^{\tilde{r}}, \tilde{h}^{\tilde{r}}),$$

$$\text{Comm}(a, r) = g^a h^r,$$

where $\bar{r}, \tilde{r}, r \xleftarrow{\$} \mathbb{F}_p$, where $\mathbb{F}_p$ is a finite field of order p , the public keys are updated using the update function $\text{Update}()$ defined in the UPK system described above, and the order amount a is protected using a commitment scheme $\text{Comm}()$. In some embodiments, the commitment scheme utilized may be a Pedersen commitment scheme disclosed in Torben Prys Pedersen, "Non-Interactive and Information-

Theoretic Secure Verifiable Secret Sharing,” Advances in Cryptology--CRYPTO 91 Procs., Lecture Notes in Computer Science, vol. 576, pp. 129-140, 1991, which is herein incorporated by reference in its entirety.

[0034] In some embodiments, the encrypted purchase order submitted to the Blockchain may include information containing an order identifier *OrderID*, the Buyer's updated public key $\overline{pk}'$, the Seller's updated public key $\widetilde{pk}'$, the commitment value of the amount $\text{Comm}(a, r)$, a settlement status (which may be set to “unsettled” if the purchase order has not been settled), and other information the Buyer may want to include. In some embodiments, the order identifier *OrderID* may include a unique identifier of the order, which may include a randomly generated universally unique identifier (UUID). In some embodiment, the settlement status may be changed to “settled” once the purchase order is settled/fulfilled.

[0035] In some embodiments, the Buyer may also generate a proof at step 306 and submit the proof to the Blockchain at step 308. In some embodiments, the proof may be a zero-knowledge proof, which 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. In some embodiments, the Seller and the Blockchain may agree to implement zero-knowledge proof such as the proof of knowledge of a discrete logarithm, also referred to as the Schnorr protocol. In some embodiments, the Seller may also construct the zero-knowledge proof using sigma protocols, which can be transformed into non-interactive zero-knowledge proofs of knowledge using Fiat-Shamir transform. Furthermore, in some embodiments, the zero-knowledge proof techniques disclosed in C. Rackoff et al., “Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack,” Advances in Cryptology--CRYPTO 91, Proceedings of the 11th Annual International Cryptology Conference on Advances in

Cryptology, pp. 433-444, 1991, which is herein incorporated by reference in its entirety, may be utilized to generate the proof. In some embodiments, the Buyer may submit the zero-knowledge proof along with a public input $x = (\overline{pk'})$ and a private input $w = (\overline{sk'})$ so that the zero-knowledge proof can be used to prove to the Blockchain that $\text{VerifyKP}(\overline{pk'}, \overline{sk'}) \equiv 1$ and $(\overline{g'})^{\overline{sk'}} \equiv h'$ for $\overline{pk'} = (\overline{g'}, \overline{h'})$. In this manner, the Buyer can prove to the Blockchain that the Buyer is in possession of the Buyer's secret key (e.g., the user submitting the proof is the Buyer itself or is authorized to do so by the Buyer).

[0036] At step 310, the Blockchain may record the encrypted purchase order submitted by the Buyer, and if the Buyer also submitted a proof, the Blockchain may verify whether the submitted proof proves to the Blockchain that the user submitting the proof is the Buyer itself or is authorized to do so by the Buyer. 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. In some embodiments, the Blockchain may record the encrypted purchase order if the Blockchain accepts the proof submitted by the Buyer. Otherwise, the Blockchain may

report an error and refuse to record the encrypted purchase order.

[0037] At step 312, the Seller may issue an invoice to the Buyer and generate an encrypted invoice, and at step 314, the Seller may submit the encrypted invoice to the Blockchain for recordation. Similar to steps 306 and 308 described above, the Seller may update the public keys of the Buyer and the Seller and encrypt the order amount a before submitting the encrypted invoice to the Blockchain. The Seller may also generate the encrypted invoice to include information containing an order identifier $OrderID$, the Buyer's updated public key $\overline{pk}'$, the Seller's updated public key $\widetilde{pk}'$, the commitment $Comm(a, r)$ of the amount specified in the invoice, the settlement status, and other information the Seller may want to include.

[0038] In some embodiments, the Seller may also generate a proof at step 312 and submit the proof to the Blockchain at step 314. In some embodiments, the Seller may submit a zero-knowledge proof along with a public input $x = (\overline{pk}')$ and a private input $w = (\widetilde{sk}')$ so that the zero-knowledge proof can be used to prove to the Blockchain that $VerifyKP(\overline{pk}', \widetilde{sk}') \equiv 1$ and $(\tilde{g}')^{\widetilde{sk}'} \equiv \tilde{h}'$ for $\overline{pk}' = (\tilde{g}', \tilde{h}')$. In this manner, the Seller can prove to the Blockchain that the Seller is in possession of the Seller's secret key (e.g., the user submitting the proof is the Seller itself or is authorized to do so by the Seller).

[0039] At step 316, the Blockchain may record the encrypted invoice submitted by the Seller, and if the Seller also submitted a proof, the Blockchain may verify whether the submitted proof proves to the Blockchain that the user submitting the proof is the Seller itself or is authorized to do so by the Seller. In some embodiments, the Blockchain may utilize one or more smart contracts executing on the Blockchain to perform the verification. In some embodiments, the Blockchain may record the encrypted invoice if the Blockchain accepts the proof submitted by the Seller.

Otherwise, the Blockchain may report an error and refuse to record the encrypted invoice.

[0040] In some embodiments, the method 300 may require the Buyer to submit the encrypted purchase order and the Seller to submit the encrypted invoice to the Blockchain. In such embodiments, the encrypted purchase order and the encrypted invoice may be utilized together to confirm/endorse the occurrence of the trade between the Buyer and the Seller. Alternatively, in some embodiments, the method 300 may require just one of the encrypted purchase order or the encrypted invoice to be submitted for a particular trade between the Seller and the Buyer. In such embodiments, the Seller may not need to submit the encrypted invoice if the Buyer has already submitted the encrypted purchase order, and vice versa. Furthermore, in some embodiments, the method 300 may specify which data entry (e.g., the encrypted purchase order or the encrypted invoice) needs to be submitted. In such embodiments, the Seller or the Buyer may need to submit the data entry as specified.

[0041] In some embodiments, a Trusted Third Party may carry out an optional step 318, in which the Trusted Third Party may submit an endorsement of the purchase order and/or the invoice to the Blockchain. In some embodiments, the Trusted Third Party may be a trusted user, including, e.g., customs, government agencies, shipping companies, banks, and the like. Upon receiving the endorsement, at step 320, the Blockchain may verify the identity of the Trusted Third Party and record the endorsement if the identity of the Trusted Third Party can be successfully verified. In some embodiments, the Blockchain may utilize one or more smart contracts executing on the Blockchain to perform the verification.

[0042] Once the Blockchain has one or more encrypted data entries recorded therein, the Seller may proceed to generate a privacy-preserving proof of its accounts

receivable. For example, the encrypted data entries may be encrypted sales data entries such as the encrypted purchase orders and the encrypted invoices described above. In some embodiments, the Seller may choose an anonymity data set $\{I_1, I_2, \dots, I_n\}$ at step 322, where I_i represents an encrypted data entry recorded on the Blockchain. In some embodiments, each I_i may have a public status on the Blockchain indicating whether it has been settled or not. In some embodiments, only unsettled orders/invoices may be included in the anonymity data set. In some embodiments, encrypted data entries associated with the Seller may constitute only a subset of the anonymity data set $\{I_1, I_2, \dots, I_n\}$, and the rest of the anonymity data set may include encrypted data entries that are not associated with the Seller (including, e.g., encrypted invoices/purchase orders associated with other sellers). In some embodiments, the anonymity data set may be required to include at least one encrypted data entry that is not associated with the Seller (including, e.g., one or more encrypted data entries associated with one or more other users). In this manner, the anonymity data set can be used to help conceal the specific data entries associated with the Seller and the total number of data entries associated with the Seller.

[0043] Also, at step 322, the Seller may calculate a commitment of the sum of the amount a_i specified in the encrypted data entries I_i that are associated with the Seller. For example, if the commitment is calculated using a Pedersen commitment scheme, then the commitment, denoted as CA , may be calculated as:

$$CA = Comm(A, r_A) = \prod_{i=1}^N (Comm(a_i, r_i))^{s_i} = g^{\sum_{i=1}^N s_i a_i} h^{\sum_{i=1}^N s_i r_i} = g^A h^{r_A}$$

where $s_i \in \{0,1\}, 1 \leq i \leq n$, may be defined as: $s_i = \begin{cases} 1 & \text{if } I_i \text{ is associated with the Seller} \\ 0 & \text{otherwise} \end{cases}$. Because the Seller knows the values of a_i and r_i

for the orders/invoices that are associated with the Seller itself, the Seller can calculate

$A = \sum_{i=1}^N s_i \cdot a_i$ and $r_A = \sum_{i=1}^N s_i \cdot r_i$. On the other hand, because a_i and r_i for orders/invoices that are not associated with the Seller will have a corresponding $s_i = 0$, the values of a_i and r_i for orders/invoices that are not associated with the Seller will be excluded from A and CA .

[0044] In some embodiments, the Seller may also generate a proof at step 322 to prove that the commitment CA is indeed a commitment of the sum of the amount specified in the encrypted data entries associated with the Seller. In some embodiments, the proof may include a zero-knowledge proof for proving to the Blockchain that CA is a commitment of the sum of the amount a_i specified in the encrypted data entries I_i that are associated with the Seller, i.e., a commitment of $\sum_{i=1}^N s_i \cdot a_i$. In some embodiments, the Seller may submit the zero-knowledge proof along with a public input $x = (CA, \{I_1, I_2, \dots, I_n\})$ and a private input $w = \left(\{s_i\}_{1 \leq i \leq n}, \{a_i\}_{1 \leq i \leq n, \&s_i=1}, r_A, \widetilde{sk}_P \right)$ so that the zero-knowledge proof can be used to prove to the Blockchain that the following relationships are true:

- (1) $CA = g^A h^{r_A}$,
- (2) $A = \sum_{i=1}^N s_i \cdot a_i$, and
- (3) $s_i = \begin{cases} 1 & \text{if } \text{VerifyKP}(\widetilde{pk}_i', \widetilde{sk}_P) \equiv 1 \\ 0 & \text{otherwise} \end{cases}$,

where equation (1) states that CA is a Pedersen commitment of A and randomness r_A , equation (2) states that A is the weighted sum of a_i , and equation (3) states that the weight for a_i is 1 if a_i corresponds to an order/invoice that is associated with the Seller and the weight for a_i is 0 if a_i corresponds to an order/invoice that is not associated with the Seller. In some embodiments, if the Seller knows a secret key $\widetilde{sk}_P$ such that $\text{VerifyKP}(\widetilde{pk}_i', \widetilde{sk}_P)$ returns 1, then the

Blockchain may recognize the Seller as the owner of the updatable public key $\widetilde{pk}_i'$ recorded in the Seller fields of invoice I_i .

[0045] At step 324, the Seller may submit the anonymity data set $\{I_1, I_2, \dots, I_n\}$, the commitment CA , and the proof (if generated) to the Blockchain, and at step 326, the Blockchain may verify whether the submitted proof is acceptable. In some embodiments, the Blockchain may utilize one or more smart contracts executing on the Blockchain to perform the verification. In some embodiments, the Blockchain may record the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA if the Blockchain accepts the proof submitted by the Seller. The Blockchain may record the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA on the Blockchain to facilitate verifications of range proofs that will be described below. On the other hand, if the Blockchain does not accept the proof submitted by the Seller, the Blockchain may report an error and refuse to record the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA .

[0046] At step 328, the Seller may generate one or more range proofs to other users, including, e.g., interested buyers, banks, financial institutions, investors, and the like, based on the commitment CA . For example, if a user, referred to as a Verifier, wants the Seller to prove that the Seller's accounts receivable A is greater than a target threshold $target$, and if the value of A is indeed greater than $target$, then the Seller may generate a range proof to prove $A > target$. It is noted that because the commitment of A , i.e., CA , has already been recorded on the Blockchain (at step 324), the Seller may not have the option to circumvent the range proof by attempting to change the value of A . Therefore, the Verifier can trust the correctness of the method 300 described herein.

[0047] At step 330, the Seller may provide the range proof to the Verifier. In

some embodiments, the Verifier may accept range proofs that are referred to as zero-knowledge range proofs, allowing the Seller to prove $A > target$ without disclosing the underlying value of A . Such range proofs are disclosed in J. Camenisch et al.: “Efficient Protocols for Set Membership and Range Proofs,” ASIACRYPT 2008, LNCS, vol. 5350, pp. 234-252, A. Poelstra et al.: “Confidential Assets,” Financial Cryptography Workshops, 2018, or B. Bunz et al.: “Bulletproofs: Short Proofs for Confidential Transactions and More,” 2018 IEEE Symposium on Security and Privacy, IEEE 2018, all of which are herein incorporated by reference in their entirety. It is to be understood that accepting zero-knowledge range proofs may allow the Seller to prove to the Verifier that $A > target$ (or $A \geq target$, if required), without revealing the actual value of A to the Verifier, effectively preserving user privacy. It is also to be understood that the range proofs may be generated through various schemes, including, but not limited to, Bulletproof scheme, Borromean Ring signatures scheme, Boneh-Boyen signatures scheme, and the like.

[0048] As described above, the method 300 allows the Seller to prove its accounts receivable is greater than or equal to a target threshold without revealing any information about its order, such as the number of orders associated with the Seller, the identities of such orders, the amount of each order associated with the Seller, and the total amount of all the orders associated with the Seller. Furthermore, by utilizing updatable public keys, the method 300 ensures that the information recorded on the Blockchain cannot be used by a third party to link the order/invoice together, thereby preserving user privacy.

[0049] FIG. 4 illustrates a flow chart of a method 400 for providing privacy-preserving proof of accounts receivable, 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.

[0050] At step 402, a node, e.g., the node 102, may receive a plurality of encrypted data entries from a plurality of users. In some embodiments, the encrypted data entries may include one or more encrypted purchase orders submitted by buyers (including, e.g., the Buyer depicted in FIG. 3) and/or one or more encrypted invoices submitted by sellers (including, e.g., the Seller depicted in FIG. 3). In some embodiments, the public keys of the buyers and the sellers may be updated when the buyers and the sellers generate the encrypted data entries. In some embodiments, each encrypted data entry may include a buyer's updated public key, a seller's updated public key, a commitment of the amount specified in the encrypted data entry, and a settlement status. In some embodiments, the public keys of the buyers and the sellers may be updated in accordance with an updatable public key system, such as the system disclosed in Fauzi et al., as described above.

[0051] At step 404, the node 102 may record the plurality of encrypted data entries on the blockchain. In some embodiments, the node 102 may also receive a proof associated with each encrypted data entry for proving to the node 102 that the user who submitted the encrypted data entry is authorized to do so. In such embodiments, the node 102 may determine (e.g., using one or more smart contracts executing on the blockchain) whether the proof associated with the encrypted data entry is acceptable, and record the encrypted data entry on the blockchain if the proof is acceptable. In some embodiments, the proof may be a zero-knowledge proof as described above.

[0052] At step 406, the node 102 may receive an anonymity data set $\{I_1, I_2, \dots, I_n\}$ from a particular user, e.g., the Seller depicted in FIG. 3. The anonymity data set may include one or more encrypted data entries associated with the particular user and one or more encrypted data entries associated with one or more other users. In some embodiments, each encrypted data entry contained in the anonymity data set $\{I_1, I_2, \dots, I_n\}$ may have an unsettled settlement status.

[0053] At step 408, the node 102 may receive a commitment CA of a sum of the amount a_i specified in the encrypted data entries associated with the particular user. In some embodiments, the node 102 may receive the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA in the same submission.

[0054] At step 410, the node 102 may record the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA on the blockchain to facilitate a verification of a range proof generated by the particular user. In some embodiments, the node 102 may receive a zero-knowledge proof for proving that the commitment CA received is of the sum of the amount a_i specified in the encrypted data entries associated with the particular user. In such embodiments, the node 102 may determine (e.g., using one or more smart contracts executing on the blockchain) whether the zero-knowledge proof is acceptable, and record the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA on the blockchain if the proof is acceptable.

[0055] In some embodiments, the particular user (e.g., the Seller depicted in FIG. 3) may generate a range proof proving that the sum of the amount a_i specified in the encrypted data entries associated with the particular user is greater than or equal to a threshold. In some embodiments, the particular user may generate the range proof as a zero-knowledge range proof. In this manner, the particular user can prove to another user (e.g., the Verifier depicted in FIG. 3) that the sum of the amount a_i

specified in the encrypted data entries associated with the particular user is greater than or equal to the threshold without revealing any information about the sum to the other user.

[0056] FIG. 5 is a block diagram of an apparatus 500 for providing privacy-preserving proof of accounts receivable, 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 determination module 504, a recording module 506, and a notification module 508.

[0057] The receiving module 502 may receive encrypted data entries from various users. If an encrypted data entry is also provided along with a proof for proving that the user who submitted the encrypted data entry is authorized to do so, then the receiving module 502 may provide the proof to the determination module 504 to determine whether the proof is acceptable. If the proof is acceptable, the determination module 504 may request the recording module 506 to record the encrypted data entry on the blockchain. Otherwise, the determination module 504 may request the notification module 508 to notify the user of the error and refuse to record the encrypted data entry on the blockchain. In some embodiments, the proof may be a zero-knowledge proof as described above.

[0058] The receiving module 502 may also receive an anonymity data set $\{I_1, I_2, \dots, I_n\}$ from a particular user, e.g., the Seller, and receive a commitment CA of a sum of the amount a_i specified in the encrypted data entries associated with the particular user. If the particular user also submits a proof for proving that the commitment CA received is of the sum of the amount a_i specified in the encrypted data entries associated with the particular user, then the receiving module 502 may provide

the proof to the determination module 504 to determine whether the proof is acceptable. If the proof is acceptable, the determination module 504 may request the recording module 506 to record the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA on the blockchain to facilitate a verification of a range proof generated by the particular user for proving that the sum of the amount a_i specified in the encrypted data entries associated with the particular user is greater than or equal to a threshold. Otherwise, the determination module 504 may request the notification module 508 to notify the user of the error and refuse to record the anonymity data set $\{I_1, I_2, \dots, I_n\}$ and the commitment CA on the blockchain.

[0059] 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.

[0060] 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.

[0061] 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.

[0062] 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.

[0063] 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).

[0064] 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.

[0065] 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.

[0066] 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.

[0067] 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 privacy-preserving proof of accounts receivable, the method comprising:

receiving a plurality of encrypted data entries from a plurality of users;

recording the plurality of encrypted data entries on a blockchain;

receiving an anonymity data set from a particular user of the plurality of users, the anonymity data set comprising at least one encrypted data entry associated with the particular user and at least one encrypted data entry associated with another user of the plurality of users;

receiving a commitment of a sum of an amount specified in the at least one encrypted data entry associated with the particular user contained in the anonymity data set, the sum excluding an amount specified in the at least one encrypted data entry associated with the other user of the plurality of users; and

recording the anonymity data set and the commitment on the blockchain to facilitate a verification of a range proof generated by the particular user for proving that the sum is greater than or equal to a threshold without revealing the sum.

2. The method of claim 1, wherein the plurality of encrypted data entries comprise at least one encrypted purchase order submitted by a buyer or at least one encrypted invoice submitted by a seller.

3. The method of claim 2, wherein a public key of the buyer is updated when an encrypted data entry associated with the buyer is generated, and a public key of the seller is updated when an encrypted data entry associated with the seller is generated.

4. The method of claim 3, wherein each particular encrypted data entry of the plurality of encrypted data entries comprises: the buyer's updated public key, the seller's updated public key, a commitment of the amount specified in the particular encrypted data entry, and a settlement status.
5. The method of claim 4, wherein each encrypted data entry of the anonymity data set has an unsettled settlement status.
6. The method of any preceding claim, further comprising:
 - receiving a proof associated with each particular encrypted data entry of the plurality of encrypted data entries; and
 - recording each particular encrypted data entry on the blockchain if the proof associated with the particular encrypted data entry is acceptable.
7. The method of claim 6, wherein the proof is a zero-knowledge proof.
8. The method of any preceding claim, further comprising:
 - receiving a zero-knowledge proof for proving that the commitment received is of the sum of the amount specified in the at least one encrypted data entry associated with the particular user.
9. The method of any preceding claim, wherein the range proof generated by the particular user for proving that the sum is greater than or equal to the threshold is a zero-knowledge range proof.

10. A device for providing privacy-preserving proof of accounts receivable, 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 9.

11. An apparatus for providing privacy-preserving proof of accounts receivable, the apparatus comprising a plurality of modules for performing the method of any of claims 1 to 9.

12. 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 9.