

TRANSACTION CONFIRMATION USING ZERO-KNOWLEDGE PROOFS

BACKGROUND

Field

[0001] The present disclosure relates generally to cryptography, zero-knowledge proofs (“ZKPs”), and blockchain technology.

Related Art

[0002] When information or data is exchanged between parties, the question arises as to whether the exchanged information may be trusted or accepted. A party receiving the exchanged information may question whether the received information is truthful and may desire a mechanism for accepting or rejecting the information. For example, the party receiving the information may wish to verify that the received information is accurate and that the sending party has not fabricated false information. Further, the receiving party may wish to verify that a third-party has not tampered with the received information prior to receipt.

[0003] In some cases, the sending party may wish to deliver information to the receiving party without exposing private underlying data used to generate the information. For example, the sending party may wish to report an average temperature measurement but may not wish to expose the individual sensor measurements from each temperature sensor. In this manner, the sending party may wish to use a mechanism that proves that the reported average temperature measurement sent to the receiving party used all of the sensor measurements and also applied the proper formula for calculating the average. One way of ensuring privacy while verifying accuracy is using a ZKP.

[0004] In other words, a party may compute a function on data that it wants to keep confidential and share the result of the function with another party. The ZKP allows a party to verify that the function was properly applied to the data, without having to share the data itself. This allows the other party to verify the integrity of the function and the accuracy of the result without knowing the underlying data.

[0005] One example of a ZKP is the zero-knowledge succinct non-interactive argument of knowledge (“zk-SNARK”). The zk-SNARK scheme may use several algorithms including a set-up phase or key generation phase; a proving algorithm; and a verification algorithm. The set-up phase may generate a random string to generate a key and then subsequently destroy the string for security. The proving algorithm generates a proof that is relatively small in size. The size of the generated proof may be independent of the runtime of the program which results in the “succinct” naming convention. The verification algorithm may verify the proof relatively quickly in a time dependent on the size of the input and independent of the length of the computation.

[0006] The zk-SNARK scheme may be useful because the succinct proofs may use relatively little memory space for a blockchain of proofs. Further, the relatively fast verification may allow multiple users that are observing the blockchain to quickly verify a large load of transactions.

[0007] The zk-SNARK cryptography may be described in the following three algorithms:

[0008] (1) $(pk, vk) = \text{SETUP}(CS)$

[0009] (2) $(\pi) = \text{PROVE}(pk, \text{public_data}, \text{private_data})$

[0010] (3) $(\text{verified}) = \text{VERIFY}(vk, \pi, \text{public_data})$

[0011] In these algorithms, “CS” represents a constraint system defining a non-probabilistic or nondeterministic polynomial statement. The constraint system may include the function sought to be proved. Public_data may be the result of the function and private_data may be the underlying data that should remain confidential. In the example above, public_data may be the average temperature measurement, and the private_data may be the individual temperature sensor values.

[0012] The SETUP algorithm generates a proving key (“pk”) and a verification key (“vk”) based on the constraint system. The SETUP algorithm may require random numbers that must be kept secret to ensure that the proof is properly and securely generated.

[0013] The PROVE algorithm generates a proof from the proving data, the public_data, and the private data. The symbol π may represent the proof itself generated using the proving key as well as public data and private data. In some implementations, the proof π may be only 300 bytes long. zk-SNARK may have the benefit of proving a succinct proof, regardless of the complexity of the underlying function.

- [0014]** The VERIFY algorithm uses the vk and the proof, π , to verify the correctness of the public data. The VERIFY algorithm may return a binary one or zero value indicating whether or not the public data is correct.
- [0015]** After receiving information and/or verifying a ZKP, however, receiving parties may further desire a manner allowing them to accept or reject the information. For example, a receiving party may wish to avoid the situation where information is published as being accepted by the receiving party without confirmation from the receiving party. The receiving party may wish to avoid a situation where a sending party delivers information to the receiving party without acceptance.
- [0016]** For example, allowing a receiving party to confirm acceptance may be useful in complying with anti-money laundering (AML) and know your customer (KYC) regulations. These regulations could in some cases prohibit receipt of items of value (such as tokens representing money, e.g. digital currency such as Ethereum and Bitcoin) from certain individuals. To comply with these regulations, there needs to be an ability for receiving party to reject a transaction.
- [0017]** In addition, requiring a receiving party to actively confirm acceptance is useful in supply chain scenarios. As packages move from one place to another, for example from a manufacturer to a distributors warehouse to a customer, there is often a person at each point that signs for the package. The signature indicates that they have received the package and assumed liability, such as risk of loss, for the package. In a system where a token represents the package, and there is no way to confirm receipt, a sender could just send the token to the receiver, even if the receiver did not actually receive or accept the physical package. In other words, the chain of custody for the package may be unclear.
- [0018]** Not only does a receiver sometimes need to confirm acceptance, in some transactions, other third parties, aside from the sender and receiver, need to provide authorization for a transaction to occur. For example, some bank accounts may require multiple signatures from multiple individuals to make a withdrawal or a withdrawal above the certain threshold.
- [0019]** In this manner, new systems and methods are needed that are computationally efficient to provide verification while still allowing receiving parties to acknowledge receipt and/or acceptance of data. Also, new methods and systems are needed that may

easily allow receiving parties to provide confirmation while maintaining cryptographic security.

BRIEF SUMMARY

[0020] Disclosed herein are system, apparatus, device, method and/or computer program product embodiments, and/or combinations and sub-combinations thereof, for confirming transactions using zero-knowledge proofs.

[0021] In an embodiment, a data receiving system may receive data from a data sending system. The data transmitted by the data sending system may represent transaction data and/or delivery data related to assets. These assets may be physical assets, such as goods, products, art, as well as securities including bonds, insurance policies, titles to assets, currency, or cryptocurrency. Upon verifying a ZKP provided by the data sending system, the data receiving system may provide a digital signature confirming the acceptance of the transaction. The digital signature may be applied in a confidential manner to conceal the identity of the data receiving system.

[0022] Receivers of cryptographic data such as asset tokens or cryptocurrency often cannot accept transactions in zero-knowledge without exposing their identity. The embodiments described herein are directed to providing a way for a receiving party of a transaction to provide a cryptographic signature acknowledging acceptance of a transaction with a sending party while maintaining confidentiality on a public system such as a blockchain.

[0023] In some embodiments, a computer-implemented method for verifying and confirming a transaction and authorizing an asset transfer may comprise receiving a digital signature and transaction data at a receiving system from a sending system. The transaction data may comprise asset data corresponding to a transaction and describing an asset to transfer, a zero-knowledge proof (“ZKP”) indicating that the sending system is capable of satisfying a condition of the transaction, a re-randomized public key corresponding to the receiving system, and a randomization value used to randomize the re-randomized public key to conceal a public key corresponding to the receiving system. The computer-implemented method may further comprise verifying the public key of the receiving system using the randomization value and verifying the ZKP to confirm satisfaction of the

condition of the transaction. In response to verifying the ZKP, the computer-implemented method may further comprise generating a re-randomized private key to indicate that the receiving system has authorized the transaction and applying the re-randomized private key and the transaction data to a hash function to generate a signed transaction container indicating an acceptance of the asset to be transferred. The computer-implemented method may further comprise publishing the signed transaction container to a blockchain.

[0024] In an embodiment, a receiving system for verifying and confirming a transaction and authorizing an asset transfer may comprise a memory and at least one processor coupled to the memory. The at least one processor may be configured to receive a digital signature and transaction data from a sending system. The transaction data may comprise asset data corresponding to a transaction and describing an asset to transfer, a zero-knowledge proof (“ZKP”) indicating that the sending system is capable of satisfying a condition of the transaction, a re-randomized public key corresponding to the receiving system, and a randomization value used to randomize the re-randomized public key to conceal a public key corresponding to the receiving system. The at least one processor may further verify the public key of the receiving system using the randomization value and verify the ZKP to confirm satisfaction of the condition of the transaction. In response to verifying the ZKP, the at least one processor may further generate a re-randomized private key to indicate that the receiving system has authorized the transaction and apply the re-randomized private key and the transaction data to a hash function to generate a signed transaction container indicating an acceptance of the asset to be transferred. The at least one processor may further publish the signed transaction container to a blockchain.

[0025] In some embodiments, a non-transitory computer-readable device is disclosed, the non-transitory computer-readable device may have instructions stored thereon that, when executed by at least one computing device, may cause the at least one computing device to perform the computer-implemented method described above.

BRIEF DESCRIPTION OF THE DRAWINGS

[0026] The accompanying drawings are incorporated herein and form a part of the specification.

- [0027]** FIG. 1A depicts a block diagram of a transaction environment, according to some embodiments.
- [0028]** FIG. 1B depicts a block diagram of a transaction environment with multiple signing systems, according to some embodiments.
- [0029]** FIG. 2 depicts a block diagram of a transaction environment displaying subsystem interfaces, according to some embodiments.
- [0030]** FIG. 3 depicts a flowchart illustrating a method for transmitting a secure transaction, according to some embodiments.
- [0031]** FIG. 4 depicts a flowchart illustrating a method for confirming a secure transaction, according to some embodiments.
- [0032]** FIG. 5 depicts an example computer system useful for implementing various embodiments.
- [0033]** In the drawings, like reference numbers generally indicate identical or similar elements. Additionally, generally, the left-most digit(s) of a reference number identifies the drawing in which the reference number first appears.

DETAILED DESCRIPTION

- [0034]** Provided herein are system, apparatus, device, method and/or computer program product embodiments, and/or combinations and sub-combinations thereof, for confirming transactions using zero-knowledge proofs. In some embodiments, a receiver may confirm acceptance of a transaction. In other embodiments, other parties may need to provide a confirmation for a transaction to occur. For example, a sender may need to get permission from other parties to send an asset. Embodiments described herein may provide this capability while keeping aspects of the transactions. For example, the transactions may be unlinkable, meaning that a receiver may not know some or all of the prior owners of the asset.
- [0035]** In an embodiment, a data sending system and a data receiving system may exchange data including transaction data. The transaction data may relate to assets such as physical assets, such as goods, products, art, as well as securities including bonds, insurance policies, currency, or cryptocurrency. A user using the data sending system may agree upon a transaction with a user using the data receiving system. The data sending

system may prepare data describing the agreed upon transaction to the data receiving system. Included in this transaction data may be a ZKP indicating that the data sending system has possession of the asset to be delivered in the transaction. The data sending system may package the ZKP along with other transaction information to the data receiving system. For example, the other transaction information may include a digital signature corresponding to the data sending system, asset identification data, asset types being exchanged, and/or asset values or quantities to be exchanged.

[0036] The transaction information may also include a re-randomized public key corresponding to the data receiving system. The data receiving system may use a public key for encryption purposes. The data sending system may re-randomize this public key to mask the data receiving system and to maintain confidentiality. The re-randomization may be performed using a randomization value. The data sending system may include the randomization value in an encrypted note to be sent to the data receiving system. Additionally, the data sending system may send a digital signature to the data receiving system. The data sending system may perform a hash operation on the transaction data to generate the digital signature corresponding to the data sending system. This digital signature may indicate that the data sending system has accepted the terms of the transaction.

[0037] The data receiving system may receive the digital signature as well as transaction data including the asset identification information, the ZKP, the re-randomized public key, and/or the randomization value. For example, the data receiving system may decrypt an encrypted note to extract the randomization value. The data receiving system may apply the randomization value using a randomization function to the data receiving system's own public key to determine whether the data receiving system is able to compute the received re-randomized public key. If so, the data receiving system may confirm that the data sending system has signed the transaction data and that the intended recipient is the data receiving system.

[0038] The data receiving system may then perform a hash operation on the transaction data including the ZKP to generate a second digital signature corresponding to the data sending party. The data receiving system may compare the second digital signature to the first digital signature received from the data sending system to verify that the second digital

signature matches the first digital signature. This comparison may aid in confirming the digital signature of the data sending system and/or the content of the transaction details.

[0039] With this confirmation of the transaction details and/or the ZKP, the data receiving system may confirm, accept, and/or acknowledge execution of the transaction. To acknowledge confirmation of the transaction, the data receiving system may provide a digital signature marking the received transaction data as being accepted. In some embodiments, the data receiving system may wish to maintain confidentiality in executing the transaction. For example, the data receiving system may wish to publish the transaction data onto a blockchain while masking the identity of the data receiving system. To provide a digital signature in this manner, the data receiving system may generate a re-randomized private key corresponding to the data receiving system. The private key may be a secret key. The data receiving system may then apply the re-randomized private key and the transaction data to a hash function to generate a signed transaction container. This signed transaction container may indicate signatures from the data receiving system and the data sending system. The data receiving system may then publish the signed transaction container to a blockchain. The publishing onto the immutable blockchain with the signature from the data receiving system may represent a confirmation of the transaction that may be verified.

[0040] Via this transmission from the data sending system to the data receiving system, parties may view and/or confirm transactions in a manner to avoid processing at an intermediary system. For example, the parties may communicate to negotiate a transaction, provide confirmations via digital signatures, and/or publish the signed transaction data to a blockchain. By executing the transaction confirmation embodiments disclosed herein, parties may avoid the unnecessary computational processing of communicating with additional systems. Further, in addition to saving these computational resources, the systems may provide trustable information while maintaining confidentiality using encrypted proof results and private keys used for signatures.

[0041] The confirmation process may also allow the parties to avoid the situation where a data sending system commits a transaction without obtaining a confirmation from the data receiving system. For example, a data sending system may attempt to declare that a transaction has occurred, such as the delivery of an asset to a data receiving system. The

data sending system may attempt to publish this transaction unilaterally to a blockchain despite disagreement from the data receiving system. The use of the confirmation process described herein may aid to avoid this issue by allowing the data receiving system to provide a confirmation of the transaction terms.

[0042] Various embodiments of these features will now be discussed with respect to the corresponding figures.

[0043] FIG. 1A depicts a block diagram of a transaction environment 100A, according to some embodiments. Transaction environment 100A may include data sending system 110, data receiving system 120, blockchain manager 130, and/or blockchain 140. Data sending system 110 and data receiving system 120 may be interchangeable and/or may be transaction systems that may participate in a transaction. While one data sending system 110 and one data receiving system 120 is depicted in FIG. 1A, other data sending systems 110 and/or other data receiving systems 120 may communicate and perform transactions. Data sending system 110 and data receiving system 120 may be referred to as transaction systems.

[0044] A transaction system may include one or more processors, memory, servers, routers, modems, and/or antennae configured to interface with another transaction system, blockchain manager 130, and/or blockchain 140. The transaction systems may communicate with each other, blockchain manager 130, and/or blockchain 140 via a local area network (LAN), a wireless network, and/or via the Internet. A transaction system may provide transaction data, parameters, digital signatures to each other, and/or commit data to blockchain 140 via blocks. In some embodiments, a transaction system may commit data and/or blocks to blockchain 140 without an intermediary system locking assets to perform a transaction.

[0045] The transaction systems may be used by different users, parties, or entities to perform transactions. For example, data sending system 110 may propose a transaction involving an exchange of assets with data receiving system 120. This proposal may occur as communications and/or messaging between data sending system 110 and data receiving system 120.

[0046] In some embodiments, data sending system 110 may propose a transaction and/or identify a proposed transaction using a marketplace and/or an exchange accessible via the

Internet. A host server and/or a website may allow transaction systems to post offers for asset transactions, view or retrieve posted offers, and/or may send notifications to transaction systems related to asset offers. Transactions systems may exchange messages, which may include counteroffers and/or modifications to proposed transactions. Data sending system 110 and data receiving system 120 may formulate agreed terms for proposed transactions.

[0047] As previously explained, data sending system 110 and data receiving system 120 may be interchangeable as systems participating in a transaction exchange. Data sending system 110 may be a transaction system sending transaction data to data receiving system 120. While the description of this specification states data sending system 110 or data receiving system 120, either transaction system may perform these operations.

[0048] Data sending system 110 may generate and/or encrypt zero-knowledge proofs (ZKPs) verifying that data sending system 110 is in possession of assets required for the proposed transaction and is committing those assets as part of the proposed transaction. This ZKP may be referred to as a “spend proof” and may indicate that an asset is available and has not already left the possession of data sending system 110. For example, the ZKP may indicate possession and commission of a type of asset (or asset identification), a quantity of the asset, and/or authorization to provide the asset.

[0049] For example, a ZKP may be generated using the zk-SNARK scheme. The zk-SNARK scheme may use several algorithms including a set-up phase or key generation phase; a proving algorithm; and a verification algorithm. The set-up phase may generate a random string to generate a key and then subsequently destroy the string for security. The proving algorithm generates a proof that is relatively small in size. The size of the generated proof may be independent of the runtime of the program which results in the “succinct” naming convention. The verification algorithm may verify the proof relatively quickly in a time dependent on the size of the input and independent of the length of the computation.

[0050] As previously explained, the zk-SNARK scheme may be useful because the succinct proofs may use relatively little memory space for a blockchain of proofs. Further, the relatively fast verification may allow multiple users that are observing the blockchain to quickly verify a large load of transactions. Data sending system 110 may generate the spend proof using the set-up phase and/or proving algorithm described below. Data sending

system 110 may generate the spend proof using the set-up phase and/or proving algorithm described below while data receiving system 120 may utilize the verification algorithm.

[0051] The following three algorithms may describe the zk-SNARK cryptography scheme:

[0052] (1) $(pk, vk) = \text{SETUP}(CS)$

[0053] (2) $(\pi) = \text{PROVE}(pk, \text{public_data}, \text{private_data})$

[0054] (3) $(\text{verified}) = \text{VERIFY}(vk, \pi, \text{public_data})$

[0055] In these algorithms, “CS” may represent a constraint system defining a non-probabilistic or nondeterministic polynomial statement. The constraint system may include the function sought to be proved. Public_data may be the result of the function and private_data may be the underlying data that should remain confidential. For data sending system 110, public_data may be an indication of satisfying a condition of a transaction while the private_data may be an asset balance corresponding to sending system 110. For example, data sending system 110 may transmit a particular amount of cryptocurrency for a transaction. The private_data may be an amount of cryptocurrency in a digital wallet corresponding to data sending system 110 while public_data may be an indication that the amount is sufficient for committing to the transaction.

[0056] The SETUP algorithm generates a proving key (“pk”) and a verification key (“vk”) based on the constraint system. The SETUP algorithm may require random numbers that must be kept secret to ensure that the proof is properly and securely generated.

[0057] The PROVE algorithm generates a proof from the proving data, the public_data, and the private data. The symbol π may represent the proof itself generated using the proving key as well as public data and private data. For data sending system 110, the output of the PROVE algorithm may be the ZKP for the transaction. In some implementations, the proof π may be only 300 bytes long. zk-SNARK may have the benefit of proving a succinct proof, regardless of the complexity of the underlying function.

[0058] Data receiving system 120 may utilize the VERIFY algorithm to verify a ZKP received from data sending system 110. The VERIFY algorithm uses the vk and the proof, π , to verify the correctness of the public data. The VERIFY algorithm may return a binary one or zero value indicating whether or not the public data is correct. For example, the VERIFY algorithm may verify the ZKP and prove to data receiving system 120 that data sending system 110 is capable of satisfying a condition of the transaction.

[0059] In addition to the spend proof, if a first party has agreed to exchange a first asset for a second asset in possession on the second party, the first party may use data sending system 110 to provide transaction data to data receiving system 120. This transaction data may include an identification or ID identifying the deal or transaction, a parameter indicating conditions that would cause a termination of the deal, the other party's payment address, the other party's mailbox identification (such as a topic or a queue), the providing party's desired asset type, value, and/or quantity, and/or the other party's desired asset type, value, and/or quantity. As will be further explained below, the transaction data may include this asset data, an identification of the first party and/or data sending system 110, the ZKP, a re-randomized public key corresponding to data receiving system 120, and/or a randomization value used to randomize a public key corresponding to data receiving system 120 to generate the re-randomized public key. The transaction data may be encrypted onto an output asset note. As will be further explained below, data receiving system 120 may verify this transaction and/or a signature corresponding to data sending system 110 to verify and/or confirm the transaction.

[0060] Data receiving system 120 may receive this transaction data from data sending system 110 participating in the transaction. In this manner, data receiving system 120 may identify the terms of the transaction and may confirm the ZKP provided by data sending system 110. Data receiving system 120 may then verify the transaction terms and the ZKP. In some embodiments, data receiving system 120 may provide a ZKP to data sending system 110 to confirm that data receiving system 120 is also able to provide the asset agreed upon for the transaction.

[0061] As will be further described with reference to FIG. 3 and FIG. 4, the parameters received by data receiving system 120 may allow data receiving system 120 to verify the terms of the transaction, verify the acceptance by data sending system 110, and/or apply a digital signature to the transaction data to signify acceptance of the terms of the transaction. Receiving system 120 may provide this signature in a confidential manner to preserve anonymity.

[0062] As previously described, one parameter transmitted by data sending system 110 is a ZKP. The ZKP may act as verification that data sending system 110 has possession of assets and is committing those assets as part of the transaction. In an embodiment, data

sending system 110 may generate a ZKP internally using a proof generator. A zero-knowledge proof may be a protocol or function applied to underlying data to indicate that the underlying data is true without revealing the underlying data itself. The result generated from the ZKP process may allow a verifier to verify that a statement provided by the prover is true. The ZKP may be used to prove whether data sending system 110 has the particular asset that has been requested by data receiving system 120 while still maintaining confidentiality and without requiring data sending system 110 to provide the asset as proof.

[0063] As previously explained, data sending system 110 and/or data receiving system 120 may use a zk-SNARK scheme to generate a ZKP and/or verify the ZKP. Data sending system 110 may use a SETUP algorithm based on a constraint system and/or a PROVE algorithm to generate the ZKP. The PROVE algorithm may generate a ZKP indicating that data sending system 110 is capable of satisfying a condition of the transaction. For example, if the transaction data sending system 110 to provide a particular number of units of cryptocurrency, the ZKP may prove that a cryptocurrency wallet corresponding to data sending system 110 contains sufficient funds to complete the transaction as well as proof that data sending system 110 is committing the particular number of units to the transaction. Committing a fewer number of units would result in an invalid ZKP when verified by data receiving system 120. In this manner, data receiving system 120 may trust that data sending system 110 is truly committing the agreed-upon asset and asset quantity.

[0064] Through using the ZKP process before actually transferring assets, confidence is ensured while avoiding the premature transfer of assets. For example, a party is able to view a ZKP from the other party before actually transferring assets to the other party or locking the assets in an escrow. In an embodiment involving cryptocurrency, the ZKP may prove that funding from data sending system 110 is sufficient and is committed to the transaction without necessarily providing data receiving system 120 with the quantity of cryptocurrency in a wallet corresponding to data sending system 110 and/or without requiring data sending system 110 to lock or escrow the amount prior to completing the transaction. Data sending system 110 may generate the ZKP and/or provide other parameters related to the transaction to data receiving system 120 including digital signatures and/or encrypted notes.

[0065] Data sending system 110 may generate a transaction data structure and/or a transaction metadata that may be used to sign the unsigned transaction data structure using the provided parameters. The transaction data structure may be private data referred to as an “asset note” or an “encrypted note” which may include the ZKP generated by data sending system 110. The encrypted note may be a data structure generated using private or public keys corresponding to data receiving system 120, a payment address corresponding to data receiving system 120, the asset types being exchanged, and/or the asset values or quantities to be exchanged. Data sending system 110 may save the asset note and/or the transaction metadata in a memory such as, for example, a database.

[0066] For example, an encrypted asset note including a ZKP proving that data sending system 110 is committing a requested asset to the transaction may be encrypted using a public key corresponding to data receiving system 120. In this manner, only data receiving system 120 may decrypt the data packet and access the encrypted asset note. Upon performing this encryption, data sending system 110 may transmit the corresponding encrypted asset note to the data receiving system 120. In some embodiments, data sending system 110 and data receiving system 120 may exchange encrypted asset notes with data receiving system 120 also producing an encrypted asset note for data sending system 110.

[0067] In some embodiments, an encrypted note may indicate ownership of an amount of a type of asset and may be created such that it belongs to an address. The encrypted note may include an asset identification indicating a particular type of asset, a note amount indicating an amount of the type of asset, and/or an address which may correspond to a particular user. Systems may be networked and may include public addresses which may conceal identities of users but may be used by parties for the transmission of assets. As previously explained, the note may correspond to an address. In an embodiment, to “spend” and/or transmit the note to data receiving system 120, data sending system 110 may use its own private or secret key (“sk”) to establish ownership of the asset in a digital wallet and/or transmit the asset. The sk may also enable a transaction system to reconstruct the history of transfers to and from a wallet. In some embodiments, the sk may be a length of 32 bytes or 256 bits. The wallet may be associated with a pair of public and private keys derived from the sk. The public key may be referred to as an “ak” while the private key may be referred to as the “ask”. As will be further explained below, these keys may be

used to sign transactions. In some embodiments, the public key may be used to identify users.

[0068] In some embodiments, a transaction system may maintain an address book, which may map real-world identities to an address. The address book may indicate the keys and/or identify the identity of secret keys. This secret identity information may be preserved as confidential to not expose underlying identities.

[0069] In addition to the ZKP, data sending system 110 transmit other transaction data to data receiving system 120. For example, data sending system 110 may transmit asset data, an identification of a sending party, and a re-randomized public key corresponding to data receiving system 120.

[0070] Data sending system 110 may generate the re-randomized public key using a randomization value such as a random number and/or a randomization algorithm. Data sending system 110 may transmit the re-randomized public key and/or the randomization value to data receiving system 120. The re-randomized public key may aid in masking the identity of data receiving system 120. For example, the public key corresponding to data receiving system 120 may be stored in an address book and/or may be a public identity known to systems in a network performing transactions. Re-randomizing the public key may mask the identity of data receiving system 120 when data receiving system 120 agrees to execute a transaction.

[0071] As previously explained, transaction systems such as data sending system 110 and/or data receiving system 120 may have an associated private key and public key corresponding to their respective wallets. Data receiving system 120 may have a public key (“ak_receiver”) along with a private key (“ask_receiver”). As previously explained, ak_receiver may be a unique identification of a wallet corresponding to data receiving system 120. Ask_receiver may be a private key used to sign data.

[0072] The re-randomized public key may be a randomization of ak_receiver and may be represented as “rk_receiver”. A randomization value “alpha” may be used to perform the re-randomization along with a randomization algorithm. Data sending system 110 may encrypt this randomization value and transmit the encrypted randomization value to data receiving system 120 with the transaction data. As will be further explained below, data

receiving system 120 may use the randomization value to verify that the transaction data was compiled using ak_receiver.

[0073] In addition to applying the randomization value to re-randomize a public key such as “ak” to a re-randomized public key “rk”, the randomization value may also be applied to randomize a secret or private key “ask” to a re-randomized private key “rsk”. The re-randomized private key, “rsk”, may be used by a sending system and/or a receiving system to sign a transaction. The signature may represent an acceptance of the transaction. For the receiving system, a value of “rsk_receiver” may be used to sign the transaction. In this manner, the receiving system may sign the transaction using the re-randomized private key to indicate an acceptance of the asset to be transferred. The sending system may similarly sign the transaction using its corresponding re-randomized private key.

[0074] In some embodiments, the transaction data may include an index value or an “I” value, which may be used for multiple signing parties or multiple parties to a transaction. FIG. 1B depicts an example embodiment with multiple signing systems where an index value may be included in the transaction data. This transaction data may be encrypted and/or may be encrypted in an encrypted asset note. In some embodiments, different parameters may be transmitted within an encrypted note while others may be transmitted separately.

[0075] In some embodiments, the transaction data may include a unique nullifier value. The nullifier value may prevent data sending system 110 from sending and/or spending the same note twice. In some embodiments, the nullifier may prevent assets from being committed twice. The nullifier may be uniquely and/or deterministically derived from a specific note. If data sending system 110 attempts to spend the same asset twice, the same nullifier would appear twice in each transaction. Data receiving system 120 may recognize the appearance of this nullifier and reject the transaction due to the nullifier appearing in a prior spend transaction.

[0076] As will be further explained below, the re-randomized public key may be the result of a randomization function or algorithm applied to a public key corresponding to data receiving system 120. Data sending system 110 may use a randomization value such as a random number to re-randomize the public key. By re-randomizing the public key of data receiving system 120, the identity of data receiving system 120 may be masked.

[0077] Further, in some embodiments, the public key of data receiving system 120 may not be included in the transaction data. Instead, data sending system 120 may compute a hash of the transaction data, which may include the re-randomized public key. Data sending system 120 may provide the randomization value to data receiving system 120 so that data receiving system 120 may verify the transaction and/or participation in the transaction. For example, data receiving system 120 may apply the randomization value to its own public key and compare the output to the received re-randomized public key. A matching result may verify that the public key corresponding to data receiving system 120 was used in preparing the transaction data. The randomization value may be provided to data receiving system 120 in an encrypted manner to allow data receiving system 120 to perform this verification. Further, the use of the public key of the data receiving system 120 as well as the re-randomized public key along with the encrypted randomization value may provide confidence that data receiving system 120. In some embodiments, the signatures may use a ZKP process to allow for verification in a zero-knowledge manner.

[0078] In some embodiments, data sending system 110 may provide one or more digital signatures to data receiving system 120. The one or more digital signatures may be a spend signature and/or a balance binding signature.

[0079] The spend signature may be generated from a hash result based on the transaction data. The spend signature may indicate an authorization to perform the spending at data sending system 110. The spend signature may be generated using the ZKP generated by data sending system 110. Data sending system 110 may use the re-randomized public key to generate the ZKP. The ZKP may verify a cryptographic key of data sending system 110. The spend signature may indicate that the data sending system 110 provides authorization to perform the transaction.

[0080] Data sending system 110 may also generate a balance binding signature. The balance binding signature may indicate a commitment to produce the balance of the transfer in the transaction. In some embodiments, the commitment may be a sum of spent amounts minus a sum of output amounts. Data sending system 110 may generate the balance binding signature using a hash of the transaction in a manner similar to the spend signature. Data sending system 110 may use the spend signature and/or the balance

binding signature to sign the transaction data and confirm that data sending system 110 will commit to performing the transaction according to the terms of the transaction data. Data sending system 110 may transmit the transaction and/or a digital signature to data receiving system 120 so that data receiving system 120 may verify the content of the transaction and/or the digital signature.

[0081] Data receiving system 120 may receive the transaction data and/or the digital signature from data sending system 110. The transaction data and/or the digital signature may be encrypted. Data receiving system 120 may decrypt this data to verify the transaction parameters, the ZKP, and/or the digital signature.

[0082] Data receiving system 120 may receive an encrypted asset note and may decrypt the encrypted asset note. Data receiving system 120 may use a decoder to perform the decryption. Data receiving system 120 may use its corresponding cryptographic key to perform the decryption. If the decryption fails, the data receiving system 120 may identify this issue as a potential error. The error could have occurred during delivery or receipt. In an embodiment, the provided encrypted asset note may have experienced tampering. For example, a malicious system may have attempted to provide a false asset note. The failure of the decryption may indicate this potential attack.

[0083] If decryption is successfully performed, data receiving system 120 may retrieve the asset note, which may include the randomization value, and perform a verification function. This verification function may verify the ZKP provided by the other party. Data receiving system 120 may verify that the other party has committed the requested asset to the transaction. For example, if data receiving system 120 has requested \$100 from data sending system 110 in exchange for a bond, data receiving system 120 may verify that data sending system 110 has actually committed this amount. Data receiving system 120 may use the parameters of the transaction, including the amount requested, in a verification function to confirm the intended result. In an embodiment, this verification may include recomputing a hash value to determine that the correct assets and/or asset quantity have been committed.

[0084] As will be further explained below, the verification function may use a signature corresponding to data sending system 110 and/or data receiving system 120 to verify the received ZKP. If the result of the verification function is an expected result in view of

the requested assets, then data receiving system 120 is able to confirm that data sending system 110 has committed the requested asset and/or asset quantity to the transaction. If the result of the verification function is not the expected result, however, data receiving system 120 may identify the verification attempt as a failure. This failure may represent data sending system 110 committing an asset or asset quantity differing from the agreed upon terms of the transaction. In this case, data receiving system 120 may reject the transaction because data receiving system 120 has detected that data sending system 110 has not committed the requested asset or asset quantity.

[0085] In the case where data receiving system 120 is able to verify the received asset note and ZKP, data receiving system 120 may agree to confirm the transaction. Confirmation of the transaction may result in the exchange of the assets and/or a posting of the transaction onto blockchain 140 by data receiving system 120. Upon verification by data receiving system 120, data receiving system 120 may apply one or more signatures to encode the transaction. Data sending system 110 may have already applied its corresponding signature when sending the transaction data to data receiving system 120. In an embodiment, data sending system 110 may sign the transaction with a signature corresponding to data receiving system 120.

[0086] To confirm the transaction, data receiving system 120 may apply a re-randomized private key to the transaction data. The re-randomized private key may be a digital signature applied by data receiving system 120. In some embodiments, the re-randomized private key may be a secret key corresponding to data receiving system 120. Data receiving system 120 may generate the re-randomized private key by applying a randomization algorithm or function to a private key. Upon generating the re-randomized private key, data receiving system 120 may apply the re-randomized private key and the transaction data to a hash function to generate a signed transaction container. This signed transaction container may include a digital signature from both data sending system 110 and data receiving system 120.

[0087] Data receiving system 120 may then publish the signed transaction container to blockchain 140 to preserve and/or confirm the transaction as well as the digital signatures. Because the transaction has been encoded using one or more signatures, the transaction may remain encoded on blockchain 140 even if the transaction is viewable by others. In

some embodiments, data receiving system 120 may use a blockchain manager 130 to publish the transaction. For example, the blockchain manager 130 may be a separate system administering blockchain 140. Data receiving system 120 may provide the signed and/or encrypted transaction container to blockchain manager 130 so that blockchain manager 130 may format the data into a block. Blockchain manager 130 may then store this data in blockchain 140. In some embodiments, data receiving system 120 may publish the signed transaction container directly to blockchain 140.

[0088] Using this signature system, data sending system 110 and data receiving system 120 may participate and sign transactions without using an intermediary system. For example, the transaction systems may directly communicate in a decentralized manner without an intermediary system. The transactions systems may exchange messages to negotiate a transaction, to exchange encrypted asset notes, and/or to provide confirmation after verifying a received ZKP. After data receiving system 120 has confirmed the ZKP and signature from data sending system 110, data receiving system 120 may sign the transaction and commit the transaction to blockchain 140.

[0089] This configuration also may preserve confidentiality while using the verification capabilities of ZKP. Further, this configuration allows for the avoidance of an intermediary escrow system in transactions while also preventing data sending system 110 from committing a transaction without a confirmation from data receiving system 120. With the expectation of a digital signature by data receiving system 120, transactions posted without the digital signature may indicate that the transaction was not accepted. In this manner, data sending system 110 may be dissuaded from making incorrect or fraudulent transactions.

[0090] FIG. 1B depicts a block diagram of a transaction environment 100B with multiple signing systems 150, according to some embodiments. Multiple signing systems 150 may represent multiple parties to a transaction. For example, multiple signing systems 150 may provide signatures to a transaction to confirm that they agree to the terms of the transaction. In some embodiments, the multiple signing systems 150 may correspond to parties exchanging assets, agents of the parties, and/or third parties, such as auditors of the transaction.

[0091] Transaction environment 100B may include data sending system 110 which may operate in a manner similar to that described with reference to FIG. 1A. Blockchain manager 130 and blockchain 140 may also operate in a similar manner. Transaction environment 100B may include signing systems 150. While two signing systems 150A, 150B may be depicted in FIG. 1B, other signing systems 150 may also communicate with data sending system 110 and/or blockchain manager 130.

[0092] The multiple signing systems 150 may operate in a manner similar to data receiving system 120. Signing systems 150 may receive transaction data from data sending system 110 and/or a digital signature corresponding to data sending system 110. The transaction data may include asset data corresponding to the transaction, a ZKP from data sending system 110, a re-randomized public key, and/or a randomization value using to re-randomize the public key. In some embodiments, the transaction data may be encrypted. In some embodiments, a subset of the transaction data may be encrypted, such as the randomization value.

[0093] In the context of the multiple signing systems 150, the transaction data may include an index value. The index value may be number indicating an output proof or ZKP to be signed. For example, the index value may provide a mapping for the multiple signing systems 150 to sign the transaction. A first index value may correspond to signing system 150A while a second index value may correspond to signing system 150B. Data sending system 110 may send these corresponding index values to the signing systems 150 with the transaction data. The signing systems 150 may then verify the transaction parameters and provide their own corresponding signatures. Signing systems 150 may then publish signed transactions to blockchain 140. Data sending system 110 and/or signing systems 150 may verify the corresponding signatures on blockchain 140 to ensure that the other parties have signed. In some embodiments, these signatures may be re-randomized versions of private keys corresponding to the different signing systems 150. This private signature may aid in masking the identity of the signing systems 150.

[0094] FIG. 2 depicts a block diagram of a transaction environment 200 displaying subsystem interfaces, according to some embodiments. Transaction environment 200 may be similar to transaction environment 100A as described with reference to FIG. 1A. For example, data sending system 210 may operate in a manner similar to data sending system

110; data receiving system 220 may operate in a manner similar to data receiving system 120; and blockchain manager 230 and blockchain 240 may be similar to blockchain manager 130 and blockchain 140.

[0095] Data sending system 210 may include hash interface 212, signature manager 214, algorithm manager 216, and/or transaction database 218. Transaction database 218 may be memory storing transaction information. Data sending system 210 and data receiving system 220 may have negotiated transaction terms and may have agreed on these terms. This negotiation may occur on or off of blockchain 240. Data sending system 210 generate a ZKP and a hash based on these transaction terms.

[0096] Algorithm manager 216 may generate a zero-knowledge proof indicating that data sending system 210 is capable of performing the transaction. For example, the ZKP may verify that data sending system 210 has sufficient funding to commit assets to a transaction. As previously explained, a ZKP may be a protocol or function applied to underlying data to indicate that the underlying data is true without revealing the underlying data itself. The result generated from the ZKP process may allow a verifier to verify that a statement provided by the prover is true. The ZKP may be used to prove whether data sending system 210 has the particular asset that has been requested by data receiving system 220 while still maintaining confidentiality and without requiring data sending system 210 to provide the asset as proof.

[0097] Signature manager 214 may generate a re-randomized version of a public key corresponding data receiving system 220. This re-randomization may be performed using a randomization algorithm and a randomization value. The randomization value may be identified using a random number generator and/or one or more random number generation algorithms. By re-randomizing the public key of data receiving system 220, the identify of data receiving system 220 may be masked for confidentiality. The re-randomized public key may also aid against tampering and may aid in verifying the correctness of the identity of data receiving system 220.

[0098] Data sending system 210 may use hash interface 212 to generate a hash of the transaction data. This hash may be used to uniquely encode the transaction and the transaction terms to provide confidentiality. Hash interface 212 may apply a hash function and/or hash algorithm to one or elements of the transaction data. For example, the

transaction data may include asset identification data, an identification of data sending system 210, the ZKP generated by algorithm manager 216, the re-randomized public key corresponding to data receiving system 220, and/or the randomization value used by signature manager 214 to generate the re-randomized public key. Hash interface 212 may apply a hash function to or generate an encrypted note including the transaction information and/or the randomization value. In some embodiments, hash interface 212 may generate a ZKP to generate the hash. This hash of the transaction data may be transaction hash which may be used to generate a digital signature corresponding to data sending system 210.

[0099] In some embodiments, the transaction hash may represent a signature corresponding to data sending system 210 and/or may be used to generate one or more signatures corresponding to data sending system 210. Data sending system 210 may also use a public key corresponding to data sending system 210 to generate this signature. For example, data sending system 210 may apply signature manager 214 to the transaction hash to generate a digital signature. As previously described, data sending system 210 may generate a spend signature and/or a balance binding signature. Signature manager 214 may generate these signatures using the transaction hash.

[0100] Data sending system 210 may transmit one or more of these signatures to data receiving system 220. Data sending system 210 may also transmit the transaction data. In some embodiments, the transaction data may be encrypted such as using an encrypted asset note.

[0101] Data receiving system 220 may receive the transaction data and/or the signature from data sending system 210. Data receiving system 220 may use hash interface 222, signature manager 224, and/or algorithm manager 216 to verify the content of the transaction, verify the signature of the data sending system 210, provide a signature confirming the transaction, and/or the publishing a signed version of the transaction to blockchain 240.

[0102] Data receiving system 220 may decrypt the transaction data and/or identify the elements of the transaction data. For example, data receiving system 220 may parse the transaction data to identify asset identification data, an identification of data sending system 210, the ZKP generated by data sending system 210, the re-randomized public key

corresponding to data receiving system 220, and/or the randomization value used by data sending system 210 to generate the re-randomized public key.

[0103] Data receiving system 220 may then use signature manager 224 to verify that the public key corresponding to data receiving system 220 was indeed used by data sending system 210 in compiling the transaction. Data receiving system 220 may perform this verification using the randomization value provided by data sending system 210. For example, data receiving system 220 may apply the randomization value to its own public key to determine whether the output matches the re-randomized public key sent by data sending system 210. This may be performed using a verification function which may apply a function incorporating the randomization function used by data sending system 210 at signature manager 214. The verification function may identify whether a match exists.

[0104] Data receiving system 220 may confirm the transaction data as well as the signature of data sending system 210. To confirm the transaction data, data receiving system 220 may identify the content of the asset identification transmitted by data sending system 210. In some embodiments, data receiving system 220 may verify the ZKP to ensure that data sending system 210 is capable of transmitting the agreed upon assets. Algorithm manager 216 may apply a ZKP verification algorithm and/or function to verify the ZKP.

[0105] Data receiving system 220 may further confirm that the signature provided by data sending system 210. Data receiving system 220 may use hash interface 222 to perform this verification. In particular, data receiving system 220 may perform a hash operation on the transaction data to generate a digital signature, which may correspond to data sending system 210. In this manner, data receiving system 220 may attempt to perform the same hash function used by data sending system 210. Using this digital signature, data receiving system 220 may compare the generated digital signature with the one received from data sending system 210. Data receiving system 220 may verify that these signatures match to verify that data sending system 210 has signed the transaction.

[0106] In response to verifying the transaction data and signature sent by data sending system 210, data receiving system 220 may generate a re-randomized private key, which may be used to sign the transaction on behalf of data receiving system 220. The re-randomized private key may differ from a public key to aid in concealing the identity of data receiving system 220. Data receiving system 220 may generate this re-randomized

private key using a randomization algorithm and/or function. Data receiving system 220 may then apply the re-randomized private key and the transaction data to a hash function to generate a signed transaction container. This hash may represent the confirmation and/or acceptance of the transaction by data receiving system 220. The hash may also provide encryption of the transaction to preserve confidentiality. Data receiving system 220 may then publish the signed transaction container to blockchain 240. In some embodiments, data receiving system 220 may publish the signed transaction container directly to blockchain 240 and/or using blockchain manager 230, which may administer blockchain 240.

[0107] By signing and/or confirming transactions in this manner, data sending system 210 and/or data receiving system 220 may sign and/or execute transactions without using an intermediary system, such as an escrow system. Further, the use of ZKPs allows for verification of asset data while the re-randomization of cryptographic keys may preserve confidentiality.

[0108] FIG. 3 depicts a flowchart illustrating a method 300 for transmitting a secure transaction, according to some embodiments. Method 300 shall be described with reference to FIG. 1A; however, method 300 is not limited to that example embodiment.

[0109] In an embodiment, data sending system 110 may utilize method 300 to compile transaction data and/or generate a digital signature. While method 300 is described with reference to data sending system 110, method 300 may be executed on any computing device, such as, for example, the computer system described with reference to FIG. 5 and/or processing logic that may comprise hardware (e.g., circuitry, dedicated logic, programmable logic, microcode, etc.), software (e.g., instructions executing on a processing device), or a combination thereof.

[0110] It is to be appreciated that not all steps may be needed to perform the disclosure provided herein. Further, some of the steps may be performed simultaneously, or in a different order than shown in FIG. 3, as will be understood by a person of ordinary skill in the art.

[0111] At 305, data sending system 110 may generate a zero-knowledge proof (ZKP) indicating that a sending party is capable of satisfying a condition of a transaction. The sending party may use data sending system 110 to compile the transaction data and/or

generate the ZKP. Data sending system 110 may maintain a digital wallet corresponding to the sending party, which may include assets. Based on transaction conditions agreed upon by the sending party and the receiving party, data sending system 110 may generate a ZKP based on the content of the digital wallet. The ZKP may indicate that a balance or status of the digital wallet may satisfy a condition of the transaction.

[0112] At 310, data sending system 110 may re-randomize a public key corresponding to a receiving party using a randomization value. The receiving party may be using data receiving system 120 to receive data. The public key corresponding to the receiving party may be known by the sending party and/or the data sending system 110. Data sending system 110 may re-randomize this public key using a randomization algorithm and the randomization value. The randomization value may be a random number and/or the output of a random number generator.

[0113] At 315, data sending system 110 may compile, as transaction data, asset data corresponding to the transaction, an identification of the sending party, the ZKP, the re-randomized public key, and/or the randomization value. The asset data may include an asset identification value such as a number or characters. In some embodiments, the asset data may include a quantity of the asset. The identification of the sending party may include an address and/or an identification corresponding to the digital wallet of the sending party. The ZKP, re-randomized public, and the randomization value may be the elements previously described. Data sending system 110 may compile the transaction data in an asset note and/or may encrypt the asset note. In some embodiments, a subset of the transaction data may be encrypted including the randomization value.

[0114] In some embodiments with multiple signing parties to a transaction, the transaction data may include an index value. As previously explained, the index value may be number indicating an output proof or ZKP to be signed. For example, the index value may provide a mapping for multiple signing systems to sign the transaction. A first index value may correspond to a first signing system while a second index value may correspond to a second signing system. Data sending system 110 may send these corresponding index values to the signing systems with the transaction data.

[0115] At 320, data sending system 110 may perform a hash operation on the transaction data to generate a digital signature corresponding to the sending party. The hash operation

may encode or encrypt the transaction data in a manner decodable by data receiving system 120. In this manner, the hash operation may preserve confidentiality where other parties may not be able to decrypt the hash of the transaction data.

[0116] In some embodiments, the hash may be generated in a manner to indicate a digital signature corresponding to the sending party. For example, the hash may be generated using identification information of the sending party. In some embodiments, the hash may be used to create a digital signature. For example, the hash may generate a spend signature and/or a balance binding signature.

[0117] The spend signature may indicate an authorization to perform the spending at data sending system 110. The spend signature may be generated using the ZKP generated by data sending system 110. Data sending system 110 may use the re-randomized public key to generate the ZKP. The ZKP may verify a cryptographic key of data sending system 110. The spend signature may indicate that the data sending system 110 provides authorization to perform the transaction.

[0118] Data sending system 110 may also generate a balance binding signature. The balance binding signature may indicate a commitment to produce the balance of the transfer in the transaction. In some embodiments, the commitment may be a sum of spent amounts minus a sum of output amounts. Data sending system 110 may generate the balance binding signature using the hash of the transaction in a manner similar to the spend signature. Data sending system 110 may use the spend signature and/or the balance binding signature to sign the transaction data and confirm that data sending system 110 will commit to performing the transaction according to the terms of the transaction data.

[0119] At 325, data sending system 110 may transmit the digital signature and the transaction data to the receiving party. Data sending system 110 may transmit the transaction and/or the digital signature to data receiving system 120 so that data receiving system 120 may verify the content of the transaction and/or the digital signature. This verification is further described with reference to FIG. 4.

[0120] FIG. 4 depicts a flowchart illustrating a method 400 for confirming a secure transaction, according to some embodiments. Method 400 shall be described with reference to FIG. 1A; however, method 400 is not limited to that example embodiment.

- [0121]** In an embodiment, data receiving system 120 may utilize method 400 to verify a digital signature, verify transaction data, authorize an asset transfer, and/or generate a signed transaction. While method 400 is described with reference to data receiving system 120, method 400 may be executed on any computing device, such as, for example, the computer system described with reference to FIG. 5 and/or processing logic that may comprise hardware (e.g., circuitry, dedicated logic, programmable logic, microcode, etc.), software (e.g., instructions executing on a processing device), or a combination thereof.
- [0122]** It is to be appreciated that not all steps may be needed to perform the disclosure provided herein. Further, some of the steps may be performed simultaneously, or in a different order than shown in FIG. 4, as will be understood by a person of ordinary skill in the art.
- [0123]** At 405, data receiving system 120 may receive and/or decrypt a first digital signature corresponding to a sending party and/or transaction data received from the sending party. The transaction data may include asset data corresponding to a transaction and describing an asset to transfer, an identification of the sending party, a ZKP indicating that the sending party is capable of satisfying a condition of the transaction, a re-randomized public key corresponding to the receiving party, and/or a randomization value used to generate the re-randomized public key.
- [0124]** In some embodiments, the transaction data may be included in an encrypted asset note. Data receiving system 120 may decrypt the asset note to extract and/or parse the transaction data. Data receiving system 120 may separate the digital signature from the transaction data.
- [0125]** At 410, data receiving system 120 may apply the randomization value and the re-randomized public key to a verification function to verify an original public key. Data receiving system 120 may verify that the original public key was used by data sending system 110 in compiling the transaction. To perform this verification, data receiving system 120 may apply the randomization value to its own public key to determine whether the output matches the re-randomized public key sent by data sending system 110. This may be performed using a verification function which may apply a function incorporating the randomization function used by data sending system 110. The verification function may identify whether a match exists between the received re-randomized public key and

the newly generated re-randomized public key based on an application of the randomization value to the original public key.

[0126] At 415, data receiving system 120 may perform a hash operation on the transaction data to generate a second digital signature corresponding to the sending party. The hash operation may aid in verifying the signature of the sending party.

[0127] Data receiving system 120 may also confirm the transaction data as well as the signature of data sending system 110. To confirm the transaction data, data receiving system 120 may identify the content of the asset identification transmitted by data sending system 110. In some embodiments, data receiving system 120 may verify the ZKP to ensure that data sending system 110 is capable of transmitting the agreed upon assets. Data receiving system 120 may apply a ZKP verification algorithm and/or function to verify the ZKP. Verifying the ZKP may confirm the satisfaction of the condition of the transaction, such as verifying that data sending system 110 has committed the correct type and/or quantity of the asset.

[0128] Data receiving system 120 may further confirm that the signature provided by data sending system 110. Data receiving system 120 may generate a hash of the transaction data to perform this verification. In particular, data receiving system 120 may perform a hash operation on the transaction data to generate the second digital signature, which may correspond to data sending system 110. In this manner, data receiving system 120 may attempt to perform the same hash function used by data sending system 110.

[0129] At 420, data receiving system 120 may compare the second digital signature to the first digital signature to verify that the second digital signature matches the first digital signature. Using this second digital signature, data receiving system 220 may compare the second digital signature with the first digital signature received from data sending system 110. Data receiving system 120 may verify that these signatures match to verify that data sending system 110 has signed the transaction. In some embodiments, data sending system 110 may have transmitted a spend signature and a balance binding signature. Data receiving system 120 may generate two signatures corresponding to each of these signatures and perform a comparison to identify two matches to verify the signatures.

[0130] At 425, data receiving system 120 may generate a re-randomized private key. Data receiving system 120 may generate this re-randomized private key in response to verifying

the re-randomized public key and/or the first digital signature provided by data sending system 110. The re-randomized private key may be a secret key corresponding to data receiving system 120. Data receiving system 120 may generate the re-randomized private key by applying a randomization algorithm or function to a private key. This re-randomized private key may aid in maintaining confidentiality of data receiving system 120 when publishing transactions to blockchain 140. Generating the re-randomized private key may indicate that data receiving system 120 has authorized the transaction.

[0131] In some embodiments, the re-randomized private key may be generated and/or verified using a ZKP or zk-SNARK scheme. For example, data receiving system 120 may use a zero-knowledge proof to conceal the identity of data receiving system 120 but to still provide confidence to verification systems that the transaction has been validly accepted and/or signed. This ZKP process may provide confidentiality but may provide confirmation that the transaction has been agreed upon and executed. The re-randomized private key may further use the private key corresponding to the digital wallet of data receiving system 120.

[0132] At 430, data receiving system 120 may apply the re-randomized private key and the transaction data to a hash function to generate a signed transaction container. This signed transaction container may include a digital signature from both data sending system 110 and data receiving system 120. Applying the re-randomized private key to the transaction data may indicate an acceptance and/or confirmation by data receiving system 120 of the transaction and/or may indicate that the asset is transferred to data receiving system 120.

[0133] At 435, data receiving system 120 may publish the signed transaction container to blockchain 140. Publishing the signed transaction container may preserve and/or confirm the transaction as well as the digital signatures. Because the transaction has been encoded using one or more signatures, the transaction may remain encoded on blockchain 140 even if the transaction is viewable by others. In some embodiments, data receiving system 120 may use a blockchain manager 130 to publish the transaction. For example, the blockchain manager 130 may be a separate system administering blockchain 140. Data receiving system 120 may provide the signed and/or encrypted transaction container to blockchain manager 130 so that blockchain manager 130 may format the data into a block. Blockchain

manager 130 may then store this data in blockchain 140. In some embodiments, data receiving system 120 may publish the signed transaction container directly to blockchain 140. Publishing the transaction to the blockchain may indicate a completion of the asset transfer.

[0134] FIG. 5 depicts an example computer system useful for implementing various embodiments.

[0135] Various embodiments may be implemented, for example, using one or more well-known computer systems, such as computer system 500 shown in FIG. 5. One or more computer systems 500 may be used, for example, to implement any of the embodiments discussed herein, as well as combinations and sub-combinations thereof.

[0136] Computer system 500 may include one or more processors (also called central processing units, or CPUs), such as a processor 504. Processor 504 may be connected to a communication infrastructure or bus 506.

[0137] Computer system 500 may also include user input/output device(s) 503, such as monitors, keyboards, pointing devices, etc., which may communicate with communication infrastructure 506 through user input/output interface(s) 502.

[0138] One or more of processors 504 may be a graphics processing unit (GPU). In an embodiment, a GPU may be a processor that is a specialized electronic circuit designed to process mathematically intensive applications. The GPU may have a parallel structure that is efficient for parallel processing of large blocks of data, such as mathematically intensive data common to computer graphics applications, images, videos, etc.

[0139] Computer system 500 may also include a main or primary memory 508, such as random access memory (RAM). Main memory 508 may include one or more levels of cache. Main memory 508 may have stored therein control logic (i.e., computer software) and/or data.

[0140] Computer system 500 may also include one or more secondary storage devices or memory 510. Secondary memory 510 may include, for example, a hard disk drive 512 and/or a removable storage device or drive 514. Removable storage drive 514 may be a floppy disk drive, a magnetic tape drive, a compact disk drive, an optical storage device, tape backup device, and/or any other storage device/drive.

- [0141]** Removable storage drive 514 may interact with a removable storage unit 518. Removable storage unit 518 may include a computer usable or readable storage device having stored thereon computer software (control logic) and/or data. Removable storage unit 518 may be a floppy disk, magnetic tape, compact disk, DVD, optical storage disk, and/ any other computer data storage device. Removable storage drive 514 may read from and/or write to removable storage unit 518.
- [0142]** Secondary memory 510 may include other means, devices, components, instrumentalities or other approaches for allowing computer programs and/or other instructions and/or data to be accessed by computer system 500. Such means, devices, components, instrumentalities or other approaches may include, for example, a removable storage unit 522 and an interface 520. Examples of the removable storage unit 522 and the interface 520 may include a program cartridge and cartridge interface (such as that found in video game devices), a removable memory chip (such as an EPROM or PROM) and associated socket, a memory stick and USB port, a memory card and associated memory card slot, and/or any other removable storage unit and associated interface.
- [0143]** Computer system 500 may further include a communication or network interface 524. Communication interface 524 may enable computer system 500 to communicate and interact with any combination of external devices, external networks, external entities, etc. (individually and collectively referenced by reference number 528). For example, communication interface 524 may allow computer system 500 to communicate with external or remote devices 528 over communications path 526, which may be wired and/or wireless (or a combination thereof), and which may include any combination of LANs, WANs, the Internet, etc. Control logic and/or data may be transmitted to and from computer system 1100 via communication path 526.
- [0144]** Computer system 500 may also be any of a personal digital assistant (PDA), desktop workstation, laptop or notebook computer, netbook, tablet, smart phone, smart watch or other wearable, appliance, part of the Internet-of-Things, and/or embedded system, to name a few non-limiting examples, or any combination thereof.
- [0145]** Computer system 500 may be a client or server, accessing or hosting any applications and/or data through any delivery paradigm, including but not limited to remote or distributed cloud computing solutions; local or on-premises software (“on-premise”

cloud-based solutions); “as a service” models (e.g., content as a service (CaaS), digital content as a service (DCaaS), software as a service (SaaS), managed software as a service (MSaaS), platform as a service (PaaS), desktop as a service (DaaS), framework as a service (FaaS), backend as a service (BaaS), mobile backend as a service (MBaaS), infrastructure as a service (IaaS), etc.); and/or a hybrid model including any combination of the foregoing examples or other services or delivery paradigms.

[0146] Any applicable data structures, file formats, and schemas in computer system 500 may be derived from standards including but not limited to JavaScript Object Notation (JSON), Extensible Markup Language (XML), Yet Another Markup Language (YAML), Extensible Hypertext Markup Language (XHTML), Wireless Markup Language (WML), MessagePack, XML User Interface Language (XUL), or any other functionally similar representations alone or in combination. Alternatively, proprietary data structures, formats or schemas may be used, either exclusively or in combination with known or open standards.

[0147] In some embodiments, a tangible, non-transitory apparatus or article of manufacture comprising a tangible, non-transitory computer useable or readable medium having control logic (software) stored thereon may also be referred to herein as a computer program product or program storage device. This includes, but is not limited to, computer system 500, main memory 508, secondary memory 510, and removable storage units 518 and 522, as well as tangible articles of manufacture embodying any combination of the foregoing. Such control logic, when executed by one or more data processing devices (such as computer system 500), may cause such data processing devices to operate as described herein.

[0148] Based on the teachings contained in this disclosure, it will be apparent to persons skilled in the relevant art(s) how to make and use embodiments of this disclosure using data processing devices, computer systems and/or computer architectures other than that shown in FIG. 5. In particular, embodiments can operate with software, hardware, and/or operating system implementations other than those described herein.

[0149] In some embodiments, a data sending system and/or a data receiving system may communicate using a digital communication network, communication infrastructure 506, etc. for example using a communication device such as communication interface 524. In

some examples, the transaction data and/or digital signature may be read from memory, obtained from a remote electronic device (such as a web server, a computerized device, etc.), and so forth. In some examples, these values may be obtained from a blockchain. In some embodiments, the data sending system and/or data receiving system may use communication infrastructure 506 and/or communication interface 524 to communicate with a blockchain manager and/or a blockchain.

[0150] It is to be appreciated that the Detailed Description section, and not any other section, is intended to be used to interpret the claims. Other sections can set forth one or more but not all exemplary embodiments as contemplated by the inventor(s), and thus, are not intended to limit this disclosure or the appended claims in any way.

[0151] While this disclosure describes exemplary embodiments for exemplary fields and applications, it should be understood that the disclosure is not limited thereto. Other embodiments and modifications thereto are possible, and are within the scope and spirit of this disclosure. For example, and without limiting the generality of this paragraph, embodiments are not limited to the software, hardware, firmware, and/or entities illustrated in the figures and/or described herein. Further, embodiments (whether or not explicitly described herein) have significant utility to fields and applications beyond the examples described herein.

[0152] Embodiments have been described herein with the aid of functional building blocks illustrating the implementation of specified functions and relationships thereof. The boundaries of these functional building blocks have been arbitrarily defined herein for the convenience of the description. Alternate boundaries can be defined as long as the specified functions and relationships (or equivalents thereof) are appropriately performed. Also, alternative embodiments can perform functional blocks, steps, operations, methods, etc. using orderings different than those described herein.

[0153] References herein to “one embodiment,” “an embodiment,” “an example embodiment,” or similar phrases, indicate that the embodiment described can include a particular feature, structure, or characteristic, but every embodiment can not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an embodiment, it would be within the

knowledge of persons skilled in the relevant art(s) to incorporate such feature, structure, or characteristic into other embodiments whether or not explicitly mentioned or described herein. Additionally, some embodiments can be described using the expression “coupled” and “connected” along with their derivatives. These terms are not necessarily intended as synonyms for each other. For example, some embodiments can be described using the terms “connected” and/or “coupled” to indicate that two or more elements are in direct physical or electrical contact with each other. The term “coupled,” however, can also mean that two or more elements are not in direct contact with each other, but yet still co-operate or interact with each other.

[0154] The breadth and scope of this disclosure should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

WHAT IS CLAIMED IS:

1. A computer-implemented method for authorizing an asset transfer, comprising:
 - receiving, at a receiving system from a sending system, a digital signature corresponding to the sending system and transaction data including:
 - asset data corresponding to a transaction and describing an asset to transfer,
 - a zero-knowledge proof indicating that the sending system is capable of satisfying a condition of the transaction,
 - a re-randomized public key corresponding to the receiving system, and
 - a randomization value used to randomize the re-randomized public key to conceal a public key corresponding to the receiving system;
 - verifying the public key of the receiving system using the randomization value;
 - verifying the zero-knowledge proof to confirm satisfaction of the condition of the transaction;
 - in response to verifying the zero-knowledge proof, generating a re-randomized private key to indicate that the receiving system has authorized the transaction;
 - applying the re-randomized private key and the transaction data to a hash function to generate a signed transaction container indicating an acceptance of the asset to be transferred; and
 - publishing the signed transaction container to a blockchain.
2. The computer-implemented method of claim 1, wherein verifying the public key further comprises:
 - applying the randomization value to the public key using a randomization algorithm; and
 - comparing a result of the randomization algorithm to the re-randomized public key to identify a match.
3. The computer-implemented method of claim 1, further comprising:
 - performing a hash operation on the transaction data to generate a second digital signature;

comparing the second digital signature to the digital signature corresponding to the sending system; and

identifying that the second digital signature matches the digital signature corresponding to the sending system.

4. The computer-implemented method of claim 1, wherein the re-randomized public key is a result of the sending system applying the randomization value to the public key of the receiving system to conceal an identity of the receiving system.
5. The computer-implemented method of claim 1, wherein the re-randomized private key conceals an identity of the receiving system.
6. The computer-implemented method of claim 1, wherein the transaction data further includes an index value indicating a multiparty transaction.
7. The computer-implemented method of claim 1, wherein the publishing further comprises:
publishing the signed transaction container to the blockchain without an escrow system locking an asset corresponding to the asset data.
8. A receiving system, comprising:
a memory; and
at least one processor coupled to the memory and configured to:
receive, from a sending system, a digital signature corresponding to the sending system and transaction data including:
asset data corresponding to a transaction and describing an asset to transfer,
a zero-knowledge proof indicating that the sending system is capable of satisfying a condition of the transaction,
a re-randomized public key corresponding to the receiving system, and
a randomization value used to randomize the re-randomized public key to conceal a public key corresponding to the receiving system;

verify the public key of the receiving system using the randomization value;
verify the zero-knowledge proof to confirm satisfaction of the condition of the transaction;

in response to verifying the zero-knowledge proof, generate a re-randomized private key to indicate that the receiving system has authorized the transaction;

apply the re-randomized private key and the transaction data to a hash function to generate a signed transaction container indicating an acceptance of the asset to be transferred; and

publish the signed transaction container to a blockchain..

9. The receiving system of claim 8, wherein to verify the public key, the at least one processor is further configured to:
 - apply the randomization value to the public key using a randomization algorithm;
 - and
 - compare a result of the randomization algorithm to the re-randomized public key to identify a match.
10. The receiving system of claim 8, wherein the at least one processor is further configured to:
 - perform a hash operation on the transaction data to generate a second digital signature;
 - compare the second digital signature to the digital signature corresponding to the sending system; and
 - identify that the second digital signature matches the digital signature corresponding to the sending system.
11. The receiving system of claim 8, wherein the at least one processor is further configured to:
 - decrypt the randomization value to extract the randomization value from the transaction data.

12. The receiving system of claim 8, wherein the re-randomized public key is a result of the sending system applying the randomization value to the public key of the receiving system to conceal an identity of the receiving system.
13. The receiving system of claim 8, wherein the re-randomized private key conceals an identity of the receiving system.
14. The receiving system of claim 8, wherein to publish the signed transaction container, the at least one processor is further configured to:
 - publish the signed transaction container to the blockchain without an escrow system locking an asset corresponding to the asset data.
15. A non-transitory computer-readable device having instructions stored thereon that, when executed by at least one computing device, cause the at least one computing device to perform operations comprising:
 - receiving, at a receiving system from a sending system, a digital signature corresponding to the sending system and transaction data including:
 - asset data corresponding to a transaction and describing an asset to transfer,
 - a zero-knowledge proof indicating that the sending system is capable of satisfying a condition of the transaction,
 - a re-randomized public key corresponding to the receiving system, and
 - a randomization value used to randomize the re-randomized public key to conceal a public key corresponding to the receiving system;
 - verifying the public key of the receiving system using the randomization value;
 - verifying the zero-knowledge proof to confirm satisfaction of the condition of the transaction;
 - in response to verifying the zero-knowledge proof, generating a re-randomized private key to indicate that the receiving system has authorized the transaction;
 - applying the re-randomized private key and the transaction data to a hash function to generate a signed transaction container indicating an acceptance of the asset to be transferred; and

publishing the signed transaction container to a blockchain.

16. The non-transitory computer-readable device of claim 15, wherein verifying the public key further comprises:
 - applying the randomization value to the public key using a randomization algorithm; and
 - comparing a result of the randomization algorithm to the re-randomized public key to identify a match.
17. The non-transitory computer-readable device of claim 15, the operations further comprising:
 - performing a hash operation on the transaction data to generate a second digital signature;
 - comparing the second digital signature to the digital signature corresponding to the sending system; and
 - identifying that the second digital signature matches the digital signature corresponding to the sending system.
18. The non-transitory computer-readable device of claim 15, wherein the re-randomized public key is a result of the sending system applying the randomization value to the public key of the receiving system to conceal an identity of the receiving system.
19. The non-transitory computer-readable device of claim 15, wherein the transaction data further includes an index value indicating a multiparty transaction.
20. The non-transitory computer-readable device of claim 15, wherein the publishing further comprises:
 - publishing the signed transaction container to the blockchain without an escrow system locking an asset corresponding to the asset data.