

MULTI-PARTY SECURITY COMPUTING METHOD AND APPARATUS, AND ELECTRONIC DEVICE

TECHNICAL FIELD

[0001] One or more embodiments of this specification are related to the field of distributed computing, and in particular, to a secure multi-party computation method and apparatus, and an electronic device.

BACKGROUND

[0002] Secure multi-party computation is a cryptography technology. A plurality of parties participating in computation may jointly perform secure computation based on respectively held inputted data by executing a secure multi-party computation algorithm, to obtain a computation result without leaking the held inputted data to other parties.

[0003] Although the secure multi-party computation algorithm can protect private data, an incorrect or even malicious algorithm implementation is still prone to leakage of private data. In an ideal scenario, the plurality of parties may develop respective secure multi-party computation software and respectively use the secure multi-party computation software developed respectively to participate in the secure multi-party computation. However, in actual applications, secure multi-party computation software usually used by the plurality of parties may be provided by one provider, so that the provider needs to prove that the provided secure multi-party computation software is trustworthy and there is no leakage of private data.

SUMMARY

[0004] This specification provides a secure multi-party computation method, applicable to any computing node deployed in a distributed network, a plurality of computing nodes being deployed in the distributed network, the plurality of computing nodes jointly participating in a secure multi-party computation based on respectively held private data, the method including:

[0005] generating a computing parameter related to held private data based on a secure multi-party computation algorithm;

[0006] transmitting the computing parameter to other computing nodes participating in the secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing

nodes participating in the secure multi-party computation; and

[0007] creating an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter.

[0008] Optionally, the method further includes:

[0009] collecting computing parameters transmitted by the other computing nodes participating in the secure multi-party computation, and performing the secure multi-party computation based on the collected computing parameters transmitted by the other computing nodes; and

[0010] creating an audit log corresponding to the computing parameters transmitted by the other computing nodes, and storing the audit log, the audit log recording description information related to the computing parameters.

[0011] Optionally, the computing parameter includes:

[0012] a random number participating in the secure multi-party computation; and/or

[0013] ciphertext data participating in the secure multi-party computation.

[0014] Optionally, the description information related to the computing parameter includes:

[0015] a random number generation rule corresponding to the random number participating in the secure multi-party computation; and/or

[0016] a key corresponding to the ciphertext data participating in the secure multi-party computation.

[0017] Optionally, the method further includes:

[0018] performing, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage; or transmitting the audit log to a third-party audit node for the audit node to perform a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage.

[0019] Optionally, the performing, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage includes:

[0020] verifying, based on the random number generation rule that is recorded in the audit log and corresponds to the random number transmitted by the computing node participating in the secure multi-party computation, whether the random number is a random number

constructed based on the private data held by the other computing nodes; and

[0021] if yes, determining that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

[0022] if no, determining that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

[0023] Optionally, the performing, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage includes:

[0024] verifying, based on the key that is recorded in the audit log and corresponds to the ciphertext data transmitted by the computing node participating in the secure multi-party computation, whether the key is a known key held by the other computing nodes; and

[0025] if yes, determining that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

[0026] if no, determining that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

[0027] This specification further provides a secure multi-party computation apparatus, applicable to any computing node deployed in a distributed network, a plurality of computing nodes being deployed in the distributed network, the plurality of computing nodes jointly participating in a secure multi-party computation based on respectively held private data, the apparatus including:

[0028] a generation module, configured to generate a computing parameter related to held private data based on a secure multi-party computation algorithm;

[0029] a transmission module, configured to transmit the computing parameter to other computing nodes participating in the secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing nodes participating in the secure multi-party computation; and

[0030] a creation module, configured to create an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter.

[0031] Optionally, the apparatus further includes:

[0032] a collection module, configured to: collect computing parameters transmitted by the other computing nodes participating in the secure multi-party computation, and perform the secure multi-party computation based on the collected computing parameters transmitted by the other computing nodes;

- [0033] the creation module being further configured to:
- [0034] create an audit log corresponding to the computing parameters transmitted by the other computing nodes, and store the audit log, the audit log recording description information related to the computing parameters.
- [0035] Optionally, the computing parameter includes:
- [0036] a random number participating in the secure multi-party computation; and/or
- [0037] ciphertext data participating in the secure multi-party computation.
- [0038] Optionally, the description information related to the computing parameter includes:
- [0039] a random number generation rule corresponding to the random number participating in the secure multi-party computation; and/or
- [0040] a key corresponding to the ciphertext data participating in the secure multi-party computation.
- [0041] Optionally, the apparatus further includes:
- [0042] an audit module, configured to: perform, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage; or transmit the audit log to a third-party audit node for the audit node to perform a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage.
- [0043] Optionally, the audit module is further configured to:
- [0044] verify, based on the random number generation rule that is recorded in the audit log and corresponds to the random number transmitted by the computing node participating in the secure multi-party computation, whether the random number is a random number constructed based on the private data held by the other computing nodes; and
- [0045] if yes, determine that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or
- [0046] if no, determine that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.
- [0047] Optionally, the audit module is further configured to:
- [0048] verify, based on the key that is recorded in the audit log and corresponds to the ciphertext data transmitted by the computing node participating in the secure multi-party computation, whether the key is a known key held by the other computing nodes; and

[0049] if yes, determine that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

[0050] if no, determine that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

[0051] This specification further provides an electronic device, applicable to a distributed network, a plurality of computing nodes being deployed in the distributed network, the plurality of computing nodes jointly participating in a secure multi-party computation based on respectively held private data, the electronic device including:

[0052] a processor; and

[0053] a memory configured to store machine-executable instructions, where

[0054] by reading and executing the machine-executable instructions that are stored in the memory and correspond to control logic of secure multi-party computation, the processor is caused to:

[0055] generate a computing parameter related to held private data based on a secure multi-party computation algorithm;

[0056] transmit the computing parameter to other computing nodes participating in the secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing nodes participating in the secure multi-party computation; and

[0057] create an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter.

[0058] According to the foregoing technical solution, the computing nodes participating in the secure multi-party computation may further create and store, after generating computing parameters based on a carried secure multi-party computation algorithm, an audit log corresponding to the computing parameters, and record description information related to the computing parameters in the audit log, so that administrative users of the computing nodes may know, by checking the audit log, detailed information about the computing parameters generated based on the secure multi-party computation algorithm. Therefore, for a user of the secure multi-party computation algorithm, audit information that is more helpful to audit the algorithm may be obtained; and for a provider of the secure multi-party computation algorithm, the foregoing secure multi-party computation algorithm may be transparent and auditable without disclosing the source code of the secure multi-party computation algorithm.

BRIEF DESCRIPTION OF THE DRAWINGS

[0059] FIG. 1 is a flowchart of a secure multi-party computation method according to an exemplary embodiment.

[0060] FIG. 2 is a schematic structural diagram of an electronic device according to an exemplary embodiment.

[0061] FIG. 3 is a block diagram of a secure multi-party computation apparatus according to an exemplary embodiment.

DETAILED DESCRIPTION OF THE INVENTION

[0062] In actual applications, for the purpose of privacy protection, secure multi-party computation algorithms are generally black box algorithms, and data transmission between computing nodes carrying a secure multi-party computation algorithm is quite nontransparent.

[0063] For example, an exchange message carrying a large number of ciphertexts and/or random numbers is generally transmitted between computing nodes carrying secure multi-party computation algorithms such as a secret sharing algorithm, a homomorphic encryption algorithm, and a garbled circuit algorithm. Even for a user of the secure multi-party computation algorithm, it is very difficult to understand the specific meanings of the ciphertexts and/or random numbers carried in the exchange message.

[0064] Therefore, for a provider of the secure multi-party computation algorithm, the provider has to disclose the source code of the provided secure multi-party computation algorithm to users or a neutral body to prove that the provided secure multi-party computation algorithm is transparent and trustworthy, to complete an audit of the algorithm.

[0065] Based on this, this specification is intended to provide a technical solution in which the provider of the secure multi-party computation algorithm can provide audit information that is more helpful to audit the algorithm for users of the algorithm without disclosing the source code of the algorithm.

[0066] During implementation, a computing node in a distributed network may generate computing parameters related to held private data based on a carried secure multi-party computation algorithm. For example, in actual applications, the generated computing parameters may include a random number participating in the secure multi-party computation; and/or ciphertext data participating in the secure multi-party computation.

[0067] According to one aspect, after the computing parameter is generated, the computing node may transmit the generated computing parameter to other computing nodes participating in the secure multi-party computation in the distributed network for the other computing nodes

to perform the secure multi-party computation based on collected computing parameters transmitted by the computing nodes participating in the secure multi-party computation.

[0068] According to another aspect, the computing node may further create an audit log corresponding to the computing parameter, record description information related to the computing parameter in the audit log, and store the created audit log.

[0069] For example, the generated computing parameter includes a random number participating in the secure multi-party computation and/or ciphertext data participating in the secure multi-party computation, and the description information related to the computing parameter may include a random number generation rule corresponding to the random number participating in the secure multi-party computation and/or key information corresponding to the ciphertext data participating in the secure multi-party computation.

[0070] According to the foregoing technical solution, the computing nodes participating in the secure multi-party computation may further create and store, after generating computing parameters based on a carried secure multi-party computation algorithm, an audit log corresponding to the computing parameters, and record description information related to the computing parameters in the audit log, so that administrative users of the computing nodes may know, by checking the audit log, detailed information about the computing parameters generated based on the secure multi-party computation algorithm. Therefore, for a user of the secure multi-party computation algorithm, audit information that is more helpful to audit the algorithm may be obtained; and for a provider of the secure multi-party computation algorithm, the foregoing secure multi-party computation algorithm may be transparent and auditable without disclosing the source code of the secure multi-party computation algorithm.

[0071] This specification is described below with reference to specific embodiments and specific application scenarios.

[0072] FIG. 1 shows a secure multi-party computation method according to an embodiment of this specification, applicable to any computing node deployed in a distributed network. A plurality of computing nodes are deployed in the distributed network. The plurality of computing nodes jointly participate in a secure multi-party computation based on respectively held private data. The secure multi-party computation method performs the following steps:

[0073] Step 102: Generate a computing parameter related to held private data based on a secure multi-party computation algorithm.

[0074] Step 104: Transmit the computing parameter to other computing nodes participating in a secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing

nodes participating in the secure multi-party computation.

[0075] Step 106: Create an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter.

[0076] The distributed network may include a data network in any form that can support secure multi-party computation and uses a distributed architecture.

[0077] For example, in a scenario, the distributed network may be a secure multi-party computation platform constructed by a device cluster (for example, a server cluster) formed by a plurality of computing devices.

[0078] In another example, in another scenario, the distributed network may be alternatively a block chain network formed by a plurality of node devices, where an online service in which the plurality of node devices need to jointly participate in a secure multi-party computation based on held private data and it needs to be ensured that the held private data is not disclosed to other node devices may be deployed in the block chain network. A specific type of the online service is not specifically limited in this specification.

[0079] The computing node may include a node device that can carry the secure multi-party computation algorithm in the distributed network.

[0080] The secure multi-party computation algorithm may include secure multi-party computation algorithms that are independently developed by the computing nodes or centrally provided by a provider. For example, in actual applications, the secure multi-party computation algorithm may, for example, correspond to secure multi-party computation software independently developed by the computing nodes or centrally provided by a provider.

[0081] The computing parameters are, for example, computing parameters generated by the computing nodes participating in the secure multi-party computation based on the carried secure multi-party computation algorithms and related to the private data held by the computing nodes; and the computing nodes need to transmit generated computing parameters to each other through an exchange message, and perform secure multi-party computation based on the collected computing parameters generated by the computing nodes.

[0082] In actual applications, the computing parameters may generally include random numbers needing to participate in a secure multi-party computation; and/or ciphertext data needing to participate in a secure multi-party computation.

[0083] It should be noted that, in actual applications, the foregoing secure multi-party computation algorithm generally includes secure multi-party computation algorithms such as a secret sharing algorithm, a homomorphic encryption algorithm, or a garbled circuit algorithm. Specific content included by the computing parameters generally depends on specific algorithm

types of the secure multi-party computation algorithms carried by the computing nodes. When the node devices carry the different algorithms listed above, the content included by the computing parameters may be somewhat different.

[0084] For example, based on the secret sharing algorithm, the computing parameter generally includes N random numbers obtained by splitting the held private data by the computing nodes, where the N random numbers include $N-1$ independently generated random numbers and an operation result (that is, an N^{th} random number) obtained by performing an operation on the $N-1$ random numbers and the held private data.

[0085] Based on the homomorphic encryption algorithm, the computing parameter generally includes ciphertext data obtained by encrypting the private data based on a held key.

[0086] Based on the garbled circuit algorithm, the computing parameter generally includes random numbers (which may also be referred to as salt values of an input value and an output value) used for respectively replacing an input value (that is, inputted data participating in computation) and an output value of a garbled circuit, and ciphertext data obtained by encrypting the salt value of outputted data by using the salt value of the input value as a key.

[0087] In this specification, a plurality of computing nodes may be deployed in a distributed network, as participants of the secure multi-party computation, the computing nodes may jointly carry the same type of secure multi-party computation algorithm to participate in a secure multi-party computation without leaking the respectively held private data.

[0088] For example, the computing nodes may jointly carry secure multi-party computation software provided by the provider of the secure multi-party computation algorithm and perform message exchange by using the secure multi-party computation software to transmit random numbers and/or ciphertext data participating in the secure multi-party computation between the computing nodes, to participate in a secure multi-party computation.

[0089] In addition to jointly carrying the same type of secure multi-party computation algorithm, an independent data source, a result storage system, and a log storage system may further be separately deployed in the computing nodes deployed in the distributed network.

[0090] The data source is used for storing and maintaining private data held by data nodes.

[0091] The result storage system is used for storing and maintaining a computation result eventually obtained by participating in a secure multi-party computation by the data nodes.

[0092] It should be noted that, in some secure multi-party computation in which only a small amount of data participates, it may be alternatively not necessary to deploy an independent data source and a result storage system in the node devices. In this case, the computing nodes may directly obtain data inputted by a user through an operation interface as the data source, and

output a final computation result of the secure multi-party computation to the user through the operation interface. The log storage system is used for storing and maintaining audit logs that are created by the data nodes and related to the computing parameters.

[0093] It should be noted that, the created audit logs may be alternatively stored in a third-party storage device or storage system. In this case, it may not be necessary to deploy an independent log storage system locally in the computing nodes. That is, in actual applications, a generation node and a storage node of an audit log may be the same or different, and this is not specifically limited in this specification.

[0094] For any target computing node participating in the secure multi-party computation, when jointly completing the secure multi-party computation with other computing nodes, the target computing node may first read private data needing to participate in the secure multi-party computation from deployed data sources.

[0095] The target computing node may then perform computation on the read private data based on the carried secure multi-party computation algorithm, to generate random numbers and/or ciphertext data related to the held private data.

[0096] Further, the target computing node may construct an exchange message based on the carried secure multi-party computation algorithm and an existing message exchange mechanism, to transmit the generated random numbers and/or ciphertext data as computing parameters to other computing nodes participating in the secure multi-party computation.

[0097] The other computing nodes may collect computing parameters transmitted, through the exchange messages, by the computing nodes participating in the secure multi-party computation, perform the secure multi-party computation based on the collected computing parameters, and store the computation result in the result storage system deployed locally.

[0098] Correspondingly, the target computing node may alternatively collect computing parameters transmitted, through the exchange messages, by the computing nodes participating in the secure multi-party computation, perform the secure multi-party computation based on the collected computing parameters, and store the computation result in the result storage system deployed locally.

[0099] It should be noted that a computation process in which the computing nodes participating in the secure multi-party computation perform the secure multi-party computation based on the collected computing parameters generally depends on a specific algorithm type of the secure multi-party computation algorithm carried in the computing nodes. Generally, computation processes involved in different types of secure multi-party computation algorithms are somewhat different.

[0100] A brief description of algorithm logic of various algorithms listed above is made below by using an example in which the secure multi-party computation algorithms carried in the computing nodes are the secret sharing algorithm, the homomorphic encryption algorithm, and the garbled circuit algorithm.

[0101] 1) Secret sharing algorithm

[0102] The secret sharing algorithm is also referred to as a secret splitting algorithm. A basic principle of the secret sharing algorithm is to split a secret (for example, a key) into a plurality of shares and deliver the shares to different people for storage. The secret can be restored only when a quantity of collected shares exceeds a threshold quantity. If the quantity of the collected shares is less than the threshold quantity, any information about the secret cannot be restored based on the shares. When the secret sharing algorithm is applied to secure multi-party computation, the threshold quantity is generally the same as a quantity of computing nodes needing to participate in the secure multi-party computation and deployed in a distributed network.

[0103] When the secret sharing algorithm is applied to the secure multi-party computation, the computing nodes respectively split inputted data into shares and exchange the shares with each other. The computing nodes then respectively perform an operation on locally collected shares to obtain a share (that is, one share of the final computation result) corresponding to a final computation result of the secure multi-party computation. Eventually, the computing nodes merge all generated shares corresponding to the final computation result to obtain the final computation result, to complete the secure multi-party computation.

[0104] If the secure multi-party computation algorithm carried in the computing nodes deployed in the distributed network is the secret sharing algorithm, after the target computing node reads private data S needing to participate in the secure multi-party computation from a deployed data source, the target computing node may first split the private data into d shares (random numbers) according to the secret sharing algorithm.

[0105] For example, based on the secret sharing algorithm, $d-1$ independent random numbers may be generated in a preset random number value space P in a random extraction manner as the first $d-1$ shares, and the shares are recorded as $S_1, S_2, S_3, \dots, S_{d-1}$.

[0106] For example, if the private data S is integer data of 64 bits, $P = 2^{64}$; and the $d-1$ independent random numbers generated eventually are all values extracted from a value space of $[0, 2^{64}-1]$.

[0107] Further, an operation may be performed on the first $d-1$ random numbers and the

private data S together, and an operation result obtained through the operation is used as a d^{th} share, which is recorded as S_d .

[0108] An operation rule for performing an operation on the $d-1$ random numbers and the private data S together generally depends on a specific secret sharing algorithm.

[0109] For example, the most common trivial additive secret sharing algorithm in the secret sharing algorithm is used as an example, and d shares obtained by splitting the private data are generally required to meet the following expression:

[0110]
$$S = S_1 + S_2 + S_3 + \dots + S_d \% P,$$

[0111] where $\%$ is the modulus operator.

[0112] Based on the foregoing expression, $S_d = S - (S_1 + S_2 + S_3 + \dots + S_{d-1}) \% P$, in this algorithm, S_d may be obtained by subtracting a modulus computation result of a sum of the first $d-1$ shares and the value space P from the private data S .

[0113] After data splitting for the private data S is completed based on the secret sharing algorithm, the target computing node may reserve one share locally, construct an exchange message based on an existing message exchange mechanism, and transmit the remaining shares to other computing nodes participating in the secure multi-party computation respectively.

[0114] The target computing node may reserve the N^{th} share locally, or may reserve any one of the first $d-1$ shares. That is, when the target computing node reserves the N^{th} share locally, only the random numbers used as the first $d-1$ shares are eventually used as the computing parameters. When the target computing node reserves any one of the first $d-1$ shares locally, the remaining random numbers used as the first $d-1$ shares and an operation result of the $N-1$ random numbers and the private data S are used as the computing parameters.

[0115] According to one aspect, other computing nodes may collect shares transmitted by the computing nodes, perform computation on the collected shares according to operation logic appointed by the computing nodes to obtain a share corresponding to the final computation result, and transmit the computation result to the other computing nodes for broadcasting. The target computing node may collect the computation results broadcast by the other computing nodes, perform computation on the collected computation results again according to the appointed operation logic to obtain a final computation result obtained by performing secure multi-party computation on the private data held by the computing nodes, and then store the final computation result in the result storage system deployed locally.

[0116] According to another aspect, the target computing node may alternatively collect the shares transmitted by the other computing nodes, perform computation on the collected shares

according to the appointed operation logic to obtain a share corresponding to the final computation result, and transmit the computation result to the other computing nodes for broadcasting. The other computing nodes may collect the computation results broadcast by the computing nodes, and perform computation on the collected computation results again according to the appointed operation logic to obtain a final result obtained by performing secure multi-party computation on the held private data S .

[0117] The operation logic appointed by the computing nodes generally depends on an actual secure multi-party computation requirement, and this is not specifically limited in this specification.

[0118] 2) Homomorphic encryption algorithm

[0119] The homomorphic encryption algorithm is a type of encryption algorithm that can perform data operation in a ciphertext state. A ciphertext of the computation result obtained by performing computation based on a cleartext may also be obtained by performing computation on the ciphertext without decryption and the use of a decryption key.

[0120] For example, an operation is performed on a homomorphic ciphertext $E(a)$ of a and a homomorphic ciphertext $E(b)$ of b to obtain a ciphertext $E(a+b)$ of $a+b$, a , b or $a+b$ is not leaked in the operation process, and a decryption key does not need to be used for decrypting the ciphertext.

[0121] If the secure multi-party computation algorithm carried in the computing nodes deployed in the distributed network is the homomorphic encryption algorithm, after the target computing node reads private data S needing to participate in the secure multi-party computation from a deployed data source, the target computing node may first generate a pair of encryption key PK and decryption key SK for homomorphic encryption according to the homomorphic encryption algorithm, encrypt the private data based on the generated encryption key PK to generate ciphertext data, construct an exchange message based on the homomorphic encryption algorithm and an existing message exchange mechanism, and respectively transmit the generated ciphertext data to other computing nodes participating in the secure multi-party computation.

[0122] According to one aspect, the other computing nodes may also use the same mode, to generate a pair of encryption key PK and decryption key SK for homomorphic encryption according to the homomorphic encryption algorithm, encrypt private data read from a locally deployed data source based on the encryption key PK to obtain ciphertext data, collect ciphertext data transmitted by the computing nodes and obtained by encrypting the respectively held private data by the computing nodes, and perform computation on collected ciphertext

data transmitted by the other computing nodes and the ciphertext data generated locally together according to appointed operation logic, to obtain ciphertext data corresponding to a final computation result obtained by performing secure multi-party computation on the private data held by the computing nodes (the ciphertext data cannot be decrypted locally by the other computing nodes).

[0123] After the ciphertext data corresponding to the final computation result is obtained, the other computing nodes may broadcast the ciphertext data corresponding to the final computation result to node devices. After the target computing node receives the ciphertext data corresponding to the final computation result, the target computing node may decrypt the received ciphertext data corresponding to the final computation result based on the decryption key SK, to obtain cleartext data of the final computation result, and then store the cleartext data of the final computation result in the result storage system deployed locally.

[0124] According to another aspect, the target computing node may alternatively collect the ciphertext data transmitted by the other computing nodes and obtained by encrypting the respectively held private data, and perform computation on the collected ciphertext data transmitted by the other computing nodes and the ciphertext data generated locally together according to the appointed operation logic, to obtain ciphertext data corresponding to the final computation result obtained by performing secure multi-party computation on the private data held by the computing nodes (the ciphertext data cannot be decrypted locally by the target computing node).

[0125] After the ciphertext data corresponding to the final computation result is obtained, the target computing node may broadcast the ciphertext data corresponding to the final computation result to the other computing nodes. After the other computing nodes receive the ciphertext data corresponding to the final computation result, the other computing nodes may decrypt the received ciphertext data corresponding to the final computation result based on the decryption key SK, to obtain cleartext data of the final computation result, and then store the cleartext data of the final computation result in the result storage system deployed locally.

[0126] It should be noted that, the operation logic appointed by the computing nodes still depends on an actual secure multi-party computation requirement, and this is not specifically limited in this specification. However, any operation logic may be decomposed into basic operations such as additive homomorphic encryption and multiplicative homomorphic encryption based on the homomorphic encryption algorithm.

[0127] 3) Garbled circuit algorithm

[0128] A principle of the garbled circuit algorithm is to simulate an application-specific

integrated circuit (ASIC) by using a computer to implement secure multi-party computation. Any secure multi-party computation task may be implemented by the ASIC designed for the task. That is, a gate circuit corresponding to the operation logic may be simulated by connecting a plurality of operation gates (basic operation gates such as AND gates, and XOR gates) according to the operation logic corresponding to the secure multi-party computation task, and an operation is then performed by using the gate circuit.

[0129] Based on the garbled circuit algorithm, the operation logic corresponding to the secure multi-party computation task is generally represented as a gate circuit, each operation path of the gate circuit is then encrypted, and the encrypted gate circuit is referred to as the garbled circuit. Secure multi-party computation is implemented by eventually performing an operation of the garbled circuit by using a computer.

[0130] When secure multi-party computation is implemented based on the garbled circuit algorithm, computing nodes participating in the secure multi-party computation are generally classified into two roles, namely, a generation party of the garbled circuit and an execution party of the garbled circuit.

[0131] For example, in one secure multi-party computation in which K computing nodes participate, K-1 computing nodes are generally used as the generation party of a circuit to generate a garbled circuit, and one remaining computing node is used as the execution party of the garbled circuit to perform an operation on the garbled circuit, to obtain a final computation result.

[0132] If the secure multi-party computation algorithm carried in the computing nodes deployed in the distributed network is the garbled circuit algorithm, the operation logic corresponding to the secure multi-party computation task may first be represented as a gate circuit.

[0133] The operation logic requires that the plurality of computing nodes jointly participate based on the respectively held private data. One computing node may only correspond to one or more operation gates in the gate circuit, and the private data held by the computing node may only be used as inputted data of one or more operation gates in the gate circuit.

[0134] Further, after the target computing node reads the private data S needing to participate in the secure multi-party computation from a deployed data source, the target computing node may use the read private data S as an input value of a corresponding operation gate in the gate circuit to perform an operation in the operation gate, to obtain an output value of the operation gate.

[0135] In actual applications, the operation logic of each operation gate in the gate circuit

may be represented and performed by using a truth table. One truth table is used for recording an output value obtained by an output path when any value is inputted into an input path of the operation gate. Each row in the truth table corresponds to one input value combination of the input path. For hiding true values of the operation paths in the operation gate in the operation of the gate circuit:

[0136] According to one aspect, the target computing node may generate random numbers based on a random function carried in the garbled circuit algorithm, to replace the input values and the output values in the truth table of the operation gate, where the random numbers obtained after the replacement are referred to as salt values.

[0137] According to another aspect, the target computing node may alternatively use the salt values corresponding to the input values of the operation gate as a key, to encrypt the salt values of the output values in the truth table of the operation gate to obtain ciphertext data, and replace the salt values of the output values in the original truth table based on the ciphertext data to obtain a garbled truth table; or the target computing node may independently create a garbled truth table based on the ciphertext data (that is, the garbled truth table only includes the ciphertext data).

[0138] According to a third aspect, the target computing node may alternatively disrupt a sequence of each row in the garbled truth table.

[0139] After the target computing node completes the garbled replacement for the truth table of the corresponding operation gate in the gate circuit in the foregoing three aspects, the garbled circuit is generated at the moment. The target computing node may construct an exchange message based on an existing message exchange mechanism of the garbled circuit. The target computing node uses the salt values corresponding to the input values of the corresponding operation gate in the gate circuit and the salt values corresponding to the input values of the operation gate as a key, and uses the ciphertext data obtained by encrypting the salt values of the output values in the truth table of the operation gate as a computing parameter, to transmit the key and the computing parameter to the computing node used as the execution party of the garbled circuit.

[0140] For example, in actual applications, when secure multi-party computation is implemented based on the garbled circuit algorithm, the garbled circuit algorithm and oblivious transfer technology are generally combined. When the target computing node transmits the computing parameter to the computing node used as the execution party of the garbled circuit, on one hand, the target computing node may generally directly transmit the garbled truth table, in the form of a cleartext, to the computing node used as the execution party of the garbled

circuit; on the other hand, the oblivious transfer technology may be used, to transmit, in a blurring transmission manner, the salt values corresponding to the input values of the operation gate corresponding to the target computing node in the gate circuit to the computing node used as the execution party of the garbled circuit. A specific implementation process of transmitting data to the computing node used as the execution party of the garbled circuit based on the oblivious transfer technology is not described in detail in this specification again. When a person skilled in the art implements the technical solution of this specification, reference may be made to the record in the related art.

[0141] After receiving the computing parameters transmitted by the computing nodes, the computing node used as the execution party of the garbled circuit may perform decryption on the ciphertext data recorded in the garbled truth table of the operation gates corresponding to the computing nodes in the gate circuit according to the salt values corresponding to the input values of the operation gates corresponding to the computing nodes in the gate circuit, to obtain the salt values of the output values of the operation gates corresponding to the computing nodes in the gate circuit.

[0142] Eventually, the computing node used as the execution party of the garbled circuit may perform an operation grade by grade based on the eventually decrypted salt values of the output values of the operation gates in the gate circuit, to obtain a salt value corresponding to a final output value of the gate circuit, and then transmit the salt value as the final computation result to other computing nodes.

[0143] After receiving the salt value corresponding to the final output value of the gate circuit, the target computing node may query the truth table, to obtain an output value corresponding to the salt value based on a correspondence between a salt value and a true output value, and store the output value as an original value of a final computation result of the secure multi-party computation in the result storage system deployed locally.

[0144] For the computing nodes deployed in the distributed network, when the secure multi-party computation process shown above is completed, the computing nodes may perform visual output on the computation result stored in the result storage system to administrative users of the computing nodes.

[0145] In this specification, an exchange message carrying a large number of ciphertexts and/or random numbers is generally transmitted between computing nodes carrying secure multi-party computation algorithms such as the secret sharing algorithm, the homomorphic encryption algorithm, and the garbled circuit algorithm. Even for a user of the secure multi-party computation algorithm, it is generally difficult to understand the specific meanings of the

ciphertexts and/or random numbers carried in the exchange message.

[0146] Based on this, in this specification, when a target node device transmits, by constructing an exchange message based on the existing message exchange mechanism of the carried secure multi-party computation algorithm, the generated computing parameters to computing nodes participating in the secure multi-party computation, the target node device may further locally create an audit log related to the transmitted computing parameters, and store the created audit log in a log storage system deployed locally.

[0147] The audit log may be, for example, used for recording description information related to the computing parameters transmitted by the computing nodes participating in the secure multi-party computation.

[0148] In a shown implementation, the computing parameter that the computing node needs to transmit to other computing nodes participating in the secure multi-party computation is generally carried in a message field in the exchange message constructed by the computing node. Therefore, the specific meaning of each field and description information related to each field in the exchange message constructed by the computing node may be recorded in the audit log for auditing.

[0149] Correspondingly, when the target node device receives the computing parameters transmitted by the other computing nodes through the constructed exchange message, the target node device may further locally create an audit log related to the computing parameters transmitted by the other computing nodes, and store the created audit log in the log storage system deployed locally.

[0150] For example, the computing nodes deployed in the distributed network generally carry the same secure multi-party computation algorithm, so that the exchange message constructed by the computing nodes generally conforms to the same message format. Therefore, after receiving the exchange message transmitted by other computing nodes, the target computing node may parse the exchange message according to a standard message format to generate a corresponding audit log, and record the specific meaning of each field in the exchange message in the audit log.

[0151] In this specification, the description information that is recorded in the audit log and is related to the computing parameter may correspond to the content included by the computing parameters.

[0152] The computing parameter transmitted by the computing node to the other computing nodes through the exchange message generally includes a random number participating in the secure multi-party computation; and/or ciphertext data participating in the secure multi-party

computation. Correspondingly, the description information that is recorded in the eventually generated audit log and is related to the computing parameter may include description information corresponding to the random number participating in the secure multi-party computation and/or description information corresponding to the ciphertext data participating in the secure multi-party computation.

[0153] Specific content of the description information is not specifically limited in this specification. In actual applications, the specific content of the description information may be determined based on an actual audit requirement of an administrative user of the computing node for a secure multi-party computation algorithm carried in the computing node.

[0154] In a shown implementation, the description information that is recorded in the eventually generated audit log and is related to the computing parameter may include a random number generation rule corresponding to the random number participating in the secure multi-party computation and/or a key corresponding to the ciphertext data participating in the secure multi-party computation.

[0155] The random number generation rule may specifically cover any form of description information related to a generation mode of the random number.

[0156] For example, the random number is randomly generated in a computing node locally, is obtained by performing an operation on a locally generated random number and other data by the computing node, is received from other computing nodes or trusted random sources, is obtained by performing an operation on random numbers received from other computing nodes or trusted random sources and other data, is obtained by performing an operation on random numbers generated locally and received random numbers by the computing node, or the like. Examples are not listed in this specification one by one.

[0157] The key corresponding to the ciphertext data participating in the secure multi-party computation may include an encryption key used for encrypting the ciphertext data.

[0158] In some scenarios, in addition to the encryption key, the audit log may further carry a decryption key used for decrypting the ciphertext data; and the audit log may even carry cleartext data of the ciphertext data.

[0159] For example, in an implementation, the description information related to the computing parameter and recorded in the audit log created by the computing node and related to the computing parameter transmitted locally to other computing nodes may be somewhat different from the description information related to the computing parameters and recorded in the audit log that is created by the computing node and related to the received computing parameters transmitted by the other computing nodes.

[0160] Due to the secure multi-party computing nodes, inputted data used by each computing node generally needs to be kept confidential from other computing nodes participating in the secure multi-party computation. Therefore, the audit log created by the computing node and related to the computing parameter transmitted by the other computing nodes may only record the encryption key related to the ciphertext data if the computing node needs to record the key corresponding to the ciphertext data. The audit log created by the computing node and related to the computing parameter transmitted locally to the other computing nodes may further record the decryption key of the ciphertext data and the cleartext data of the ciphertext data in addition to the encryption key of the ciphertext data.

[0161] In this way, an administrative user of the computing node may further verify, after the ciphertext data generated locally as a computing parameter by the computing node is decrypted based on the decryption key recorded in the audit log, whether decrypted cleartext data of the computing parameter is consistent with the cleartext data of the computing parameter recorded in the audit log, and audit whether the secure multi-party computation algorithm carried in the computing node maliciously adds private data to the transmitted computing parameter without the user's knowledge, thereby improving the auditability of the algorithm.

[0162] In this specification, the computing nodes deployed in the distributed network may jointly use a secure multi-party algorithm provided by a provider of secure multi-party computation algorithms. Although the secure multi-party algorithm is an algorithm used for protecting private data, an incorrect or even malicious algorithm implementation is still prone to leakage of private data.

[0163] For example, in a scenario, a malicious secure multi-party computation algorithm may not use, according to a design of the algorithm, a random number randomly generated as a key to encrypt the private data needing to be transmitted to other computing nodes, but instead use a known key of other parties to encrypt private data needing to be transmitted to other computing nodes without a user's knowledge, to allow a receiver to obtain the private data held by the other computing nodes by using the known key to decrypt the data.

[0164] In another example, in another scenario, a malicious secure multi-party computation algorithm may construct a pseudo-random number based on private data held by the computing node without a user's knowledge, and transmit the private data to other computing nodes by using the pseudo-random number.

[0165] In this specification, the computing nodes deployed in the distributed network may perform a privacy audit on the secure multi-party algorithm carried in the computing nodes

based on the audit log stored in the log storage system (or may only perform a privacy audit on the secure multi-party algorithm carried in this node or the secure multi-party algorithm carried in a specified computing node in the computing nodes), to determine whether the computing parameters transmitted by the computing nodes contain the risk of privacy leakage.

[0166] Certainly, in actual applications, the privacy audit on the secure multi-party algorithm carried in the computing nodes may be alternatively performed by a third-party audit node. In this case, the computing nodes deployed in the distributed network may transmit the audit log in the log storage system to the audit node (for example, the audit log may be transmitted actively or the audit node may actively invoke the audit log), and the third-party audit node performs, based on the received audit log, a privacy audit on the secure multi-party algorithm carried in the computing nodes, to determine whether the computing parameters transmitted by the computing nodes contain a risk of privacy leakage.

[0167] For example, in a scenario, the computing nodes carry a secure multi-party computation algorithm provided by the provider, and audit nodes are independently deployed by operators of the computing nodes, where the audit nodes are used for performing a privacy audit on the secure multi-party computation algorithm provided by the provider.

[0168] The audit logic of performing a privacy audit on the secure multi-party algorithm carried in the computing nodes depends on specific content of the description information related to the computing parameter and recorded in the audit log.

[0169] In a shown implementation, for example, the computing parameter is a random number participating in the secure multi-party computation, and the description information related to the computing parameter is a random number generation rule corresponding to the random number participating in the secure multi-party computation.

[0170] In this scenario, the computing node may read an audit log corresponding to the random number transmitted as a computing parameter by the computing nodes participating in the secure multi-party computation from a log storage system deployed locally, and verify, based on a random number generation rule that is recorded in the read audit log and corresponds to the random number, whether the random number is a random number constructed based on private data held by the other computing nodes.

[0171] For example, in actual applications, a digest value (for example, a hash value) of data that is recorded in the random number generation rule and used to generate the random number may be computed, and the computed digest value is compared with digest values of private data held by the other computing nodes, to determine whether the random number is a pseudo-random number constructed based on the private data held by the other computing nodes.

[0172] If it is determined through verification that the random number is a random number constructed based on the private data held by the other computing nodes, the random number used as the computing parameter contains the risk of privacy leakage; or otherwise, if it is determined through verification that a data source generating the random number is a valid data source, and the random number is not a random number constructed based on the private data held by the other computing nodes, the random number used as the computing parameter does not contain the risk of privacy leakage.

[0173] In this way, the computing node may determine whether the random number used as the computing parameter is a pseudo-random number constructed based on the private data held by the other computing nodes by performing a simple audit on the audit log in the log storage system deployed locally, so that the risk of privacy leakage can be found in time.

[0174] In another shown implementation, using an example in which the computing parameters are ciphertext data participating in the secure multi-party computation; and the description information related to the computing parameters is a key corresponding to the ciphertext data participating in the secure multi-party computation:

[0175] In this scenario, the computing node may read an audit log corresponding to the ciphertext data transmitted as a computing parameter by the computing nodes participating in the secure multi-party computation from a log storage system deployed locally, and verify, based on the key that is recorded in the read audit log and corresponds to the ciphertext data, whether the key is a known key held by the other computing nodes.

[0176] For example, in actual applications, the key that is recorded in the audit log and corresponds to the ciphertext data may be compared with known keys held by the other computing nodes, to determine whether the key corresponding to the ciphertext data is a known key held by the other computing nodes.

[0177] If it is determined through verification that the key corresponding to the ciphertext data is a known key held by the other computing nodes, the random number used as the computing parameter contains the risk of privacy leakage; or otherwise, if it is determined through verification that the key corresponding to the ciphertext data does not match the known keys held by the other computing nodes, the random number used as the computing parameter does not contain the risk of privacy leakage.

[0178] In this way, the computing node may determine whether the key corresponding to the ciphertext data used as a computing parameter is a known key held by the other computing nodes by performing a simple audit on the audit log in the log storage system deployed locally, so that the risk of privacy leakage can be found in time.

[0179] Certainly, in some scenarios, if the audit log further carries a decryption key used for decrypting the ciphertext data and cleartext data of the ciphertext data in addition to the encryption key, in this case, the computing node may further perform a security verification on the ciphertext data based on the decryption key corresponding to the ciphertext data and the cleartext data of the ciphertext data that are recorded in the read audit log, to determine whether private data is maliciously added to the ciphertext data by a carried secure multi-party computation algorithm.

[0180] For example, in actual applications, after the ciphertext data generated locally as a computing parameter by the computing node is decrypted based on the decryption key recorded in the audit log, the computing node may further verify whether the decrypted cleartext data of the computing parameter is consistent with the cleartext data of the computing parameter recorded in the audit log, and audit whether the secure multi-party computation algorithm carried in the computing node maliciously adds private data to the ciphertext data transmitted as a computing parameter without the user's knowledge. If they are consistent, it indicates that the implementation logic of the secure multi-party computation algorithm that is carried in the computing node and is provided by a provider is the same as the design of the secure multi-party computation algorithm, and the ciphertext data does not contain the risk of privacy leakage. If they are not consistent, it indicates that the secure multi-party computation algorithm that is carried in the computing node and is provided by an algorithm provider may maliciously add private data to the transmitted computing parameter without the user's knowledge, and in this case, the ciphertext data contains the risk of privacy leakage.

[0181] In this way, the computing node may determine, by performing a simple audit on the audit log in the log storage system deployed locally, whether private data is maliciously added to the ciphertext data used as a computing parameter by a carried secure multi-party algorithm, so that the risk of privacy leakage can be found in time.

[0182] According to the foregoing technical solution, the computing nodes participating in the secure multi-party computation may further create and store, after generating computing parameters based on a carried secure multi-party computation algorithm, an audit log corresponding to the computing parameters, and record description information related to the computing parameters in the audit log, so that administrative users of the computing nodes may know, by checking the audit log, detailed information about the computing parameters generated based on the secure multi-party computation algorithm. Therefore, for a user of the secure multi-party computation algorithm, audit information that is more helpful to audit the algorithm may be obtained; and for a provider of the secure multi-party computation algorithm,

the foregoing secure multi-party computation algorithm may be transparent and auditable without disclosing the source code of the secure multi-party computation algorithm.

[0183] Corresponding to the foregoing method embodiment, this specification further provides an embodiment of a secure multi-party computation apparatus. The embodiment of the secure multi-party computation apparatus in this specification may be applicable to an electronic device. The apparatus embodiment may be implemented by software, or by hardware or a combination of software and hardware. Using a software implementation as an example, as a logical apparatus, the apparatus is formed by reading corresponding computer program instructions from a non-transitory storage into a memory by a processor of an electronic device where the apparatus is located. On a hardware level, as shown in FIG. 2, which is a hardware structural diagram of an electronic device in which a secure multi-party computation apparatus according to this specification is located, the electronic device in which the apparatus is located in the embodiment not only includes a processor, a memory, a network interface, and a non-transitory storage shown in FIG. 2, but usually may also include other hardware according to actual functions of the electronic device. Details are not described herein again.

[0184] FIG. 3 is a block diagram of a secure multi-party computation apparatus according to an exemplary embodiment of this specification.

[0185] Referring to FIG. 3, the secure multi-party computation apparatus 30 may be applicable to the electronic device shown in FIG. 2, and includes a generation module 301, a transmission module 302, and a creation module 303.

[0186] The generation module 301 is configured to generate a computing parameter related to held private data based on a secure multi-party computation algorithm.

[0187] The transmission module 302 is configured to transmit the computing parameter to other computing nodes participating in the secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing nodes participating in the secure multi-party computation.

[0188] The creation module 303 is configured to create an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter.

[0189] In this embodiment, the apparatus 30 further includes:

[0190] a collection module 304 (not shown in FIG. 3), configured to: collect computing parameters transmitted by the other computing nodes participating in the secure multi-party computation, and perform the secure multi-party computation based on the collected

computing parameters transmitted by the other computing nodes; and

[0191] the creation module 303 is further configured to:

[0192] create an audit log corresponding to the computing parameters transmitted by the other computing nodes, and store the audit log, the audit log recording description information related to the computing parameters.

[0193] In this embodiment, the computing parameter includes:

[0194] a random number participating in the secure multi-party computation; and/or

[0195] ciphertext data participating in the secure multi-party computation.

[0196] In this embodiment, the description information related to the computing parameter includes:

[0197] a random number generation rule corresponding to the random number participating in the secure multi-party computation; and/or

[0198] a key corresponding to the ciphertext data participating in the secure multi-party computation.

[0199] In this embodiment, the apparatus 30 further includes:

[0200] an audit module 305 (not shown in FIG. 3), configured to: perform, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage; or

[0201] transmit the audit log to a third-party audit node for the audit node to perform a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage.

[0202] In this embodiment, the audit module 305 is further configured to:

[0203] verify, based on the random number generation rule that is recorded in the audit log and corresponds to the random number transmitted by the computing node participating in the secure multi-party computation, whether the random number is a random number constructed based on the private data held by the other computing nodes; and

[0204] if yes, determine that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

[0205] if no, determine that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

[0206] In this embodiment, the audit module 305 is further configured to:

[0207] verify, based on the key that is recorded in the audit log and corresponds to the

ciphertext data transmitted by the computing node participating in the secure multi-party computation, whether the key is a known key held by the other computing nodes; and

[0208] if yes, determine that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

[0209] if no, determine that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

[0210] For details about the implementation processes of the functions and effects of the modules in the foregoing apparatus, refer to the implementation processes of the corresponding steps in the foregoing method. Details are not described herein again.

[0211] The apparatus embodiment basically corresponds to the method embodiment, and therefore for related parts, reference may be made to partial descriptions in the method embodiment. The described apparatus embodiment is merely an example. The modules described as separate parts may or may not be physically separate, and parts displayed as modules may or may not be physical modules, may be located in one position, or may be distributed on a plurality of network modules. A part or all of the modules may be selected according to actual requirements to achieve the objectives of the solutions of this specification. A person of ordinary skill in the art may understand and implement the solutions without creative efforts.

[0212] The system, apparatus, module, or unit described in the foregoing embodiment may be, for example, implemented by a computer chip or entity, or a product with a specified function. A typical implementation device is a computer, and a specific form of the computer may be a personal computer, a laptop computer, a cellular phone, a camera phone, a smartphone, a personal digital assistant, a media player, a navigation device, an email device, a game console, a tablet computer, a wearable device, or a combination of any of these devices.

[0213] Corresponding to the foregoing method embodiment, this specification further provides an embodiment of an electronic device. The electronic device includes a processor and a memory configured to store machine-executable instructions, where the processor and the memory are generally connected by an internal bus. In other possible implementations, the device may further include an external interface for communication with other devices or components.

[0214] In this embodiment, by reading and executing the machine-executable instructions that are stored in the memory and correspond to control logic of secure multi-party computation, the processor is caused to:

[0215] generate a computing parameter related to held private data based on a secure multi-

party computation algorithm;

[0216] transmit the computing parameter to other computing nodes participating in the secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing nodes participating in the secure multi-party computation; and

[0217] create an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter.

[0218] In this embodiment, by reading and executing the machine-executable instructions that are stored in the memory and correspond to control logic of secure multi-party computation, the processor is caused to:

[0219] collect computing parameters transmitted by the other computing nodes participating in the secure multi-party computation, and perform the secure multi-party computation based on the collected computing parameters transmitted by the other computing nodes; and

[0220] create an audit log corresponding to the computing parameters transmitted by the other computing nodes, and store the audit log, the audit log recording description information related to the computing parameters.

[0221] In this embodiment, by reading and executing the machine-executable instructions that are stored in the memory and correspond to control logic of secure multi-party computation, the processor is caused to:

[0222] perform, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage; or

[0223] transmit the audit log to a third-party audit node for the audit node to perform a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage.

[0224] In this embodiment, by reading and executing the machine-executable instructions that are stored in the memory and correspond to control logic of secure multi-party computation, the processor is caused to:

[0225] verify, based on the random number generation rule that is recorded in the audit log and corresponds to the random number transmitted by the computing node participating in the secure multi-party computation, whether the random number is a random number constructed based on the private data held by the other computing nodes; and

[0226] if yes, determine that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

[0227] if no, determine that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

[0228] In this embodiment, by reading and executing the machine-executable instructions that are stored in the memory and correspond to control logic of secure multi-party computation, the processor is caused to:

[0229] verify, based on the key that is recorded in the audit log and corresponds to the ciphertext data transmitted by the computing node participating in the secure multi-party computation, whether the key is a known key held by the other computing nodes; and

[0230] if yes, determine that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

[0231] if no, determine that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

[0232] Other embodiments of this specification will be apparent to a person skilled in the art from consideration of the specification and practice of the present application disclosed here. The present disclosure is intended to cover any variations, uses, or adaptations of the present disclosure following the general principles thereof and including such departures from the present disclosure as come within known or customary practice in the art. It is intended that the specification and examples be considered as exemplary only, with a true scope and spirit of the present disclosure being indicated by the following claims.

[0233] It will be appreciated that the present disclosure is not limited to the exact construction that has been described above and illustrated in the accompanying drawings, and that various modifications and changes can be made without departing from the scope thereof. It is intended that the scope of this specification only be limited by the appended claims.

[0234] The foregoing descriptions are merely exemplary embodiments of the present disclosure, but are not intended to limit the present disclosure. Any modification, equivalent replacement, or improvement made without departing from the spirit and principle of the present disclosure should fall within the protection scope of the present disclosure.

CLAIMS

What is claimed is:

1. A secure multi-party computation method, applicable to any computing node deployed in a distributed network, a plurality of computing nodes being deployed in the distributed network, the plurality of computing nodes jointly participating in a secure multi-party computation based on respectively held private data, the method comprising:

generating a computing parameter related to held private data based on a secure multi-party computation algorithm, wherein the computing parameter comprises ciphertext data participating in the secure multi-party computation;

transmitting the computing parameter to other computing nodes participating in the secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing nodes participating in the secure multi-party computation;

creating an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter, wherein the description information comprises a key corresponding to the ciphertext data participating in the secure multi-party computation; and

transmitting the audit log to a third-party audit node for the audit node to perform a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage by:

verifying, based on the key that is recorded in the audit log and corresponds to the ciphertext data transmitted by the computing node participating in the secure multi-party computation and without decrypting the ciphertext data, whether the key is a known key held by the other computing nodes; and

if yes, determining that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

if no, determining that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

2. The method according to claim 1, further comprising:

collecting computing parameters transmitted by the other computing nodes participating in the secure multi-party computation, and performing the secure multi-party computation

based on the collected computing parameters transmitted by the other computing nodes; and
creating an audit log corresponding to the computing parameters transmitted by the other computing nodes, and storing the audit log, the audit log recording description information related to the computing parameters.

3. The method according to claim 1 or 2, wherein the computing parameter comprises:
a random number participating in the secure multi-party computation.

4. The method according to claim 3, wherein the description information related to the computing parameter comprises:
a random number generation rule corresponding to the random number participating in the secure multi-party computation.

5. The method according to claim 4, further comprising:
performing, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage.

6. The method according to claim 5, wherein the performing, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage comprises:

verifying, based on the random number generation rule that is recorded in the audit log and corresponds to the random number transmitted by the computing node participating in the secure multi-party computation, whether the random number is a random number constructed based on the private data held by the other computing nodes; and

if yes, determining that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

if no, determining that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

7. A secure multi-party computation apparatus, applicable to any computing node deployed in a distributed network, a plurality of computing nodes being deployed in the

distributed network, the plurality of computing nodes jointly participating in a secure multi-party computation based on respectively held private data, the apparatus comprising:

- a generation module, configured to generate a computing parameter related to held private data based on a secure multi-party computation algorithm, wherein the computing parameter comprises ciphertext data participating in the secure multi-party computation;

- a transmission module, configured to transmit the computing parameter to other computing nodes participating in the secure multi-party computation for the other computing nodes to perform the secure multi-party computation based on collected computing parameters transmitted by the computing nodes participating in the secure multi-party computation;

- a creation module, configured to create an audit log corresponding to the computing parameter, the audit log recording description information related to the computing parameter, wherein the description information comprises a key corresponding to the ciphertext data participating in the secure multi-party computation; and

- an audit module, configured to: transmit the audit log to a third-party audit node for the audit node to perform a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage by:

- verifying, based on the key that is recorded in the audit log and corresponds to the ciphertext data transmitted by the computing node participating in the secure multi-party computation and without decrypting the ciphertext data, whether the key is a known key held by the other computing nodes; and

- if yes, determining that the computing parameter transmitted by the computing node contains the risk of privacy leakage; or

- if no, determining that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

8. The apparatus according to claim 7, further comprising:

- a collection module, configured to: collect computing parameters transmitted by the other computing nodes participating in the secure multi-party computation, and perform the secure multi-party computation based on the collected computing parameters transmitted by the other computing nodes;

- the creation module being further configured to:

- create an audit log corresponding to the computing parameters transmitted by the other computing nodes, and store the audit log, the audit log recording description information

related to the computing parameters.

9. The apparatus according to claim 7 or 8, wherein the computing parameter comprises:
a random number participating in the secure multi-party computation.

10. The apparatus according to claim 9, wherein the description information related to the computing parameter comprises:

a random number generation rule corresponding to the random number participating in the secure multi-party computation.

11. The apparatus according to claim 10, further comprising:

an audit module, configured to: perform, based on the audit log, a privacy audit on the computing parameter transmitted by the computing node participating in the secure multi-party computation, to determine whether the computing parameter transmitted by the computing node contains a risk of privacy leakage.

12. The apparatus according to claim 11, wherein the audit module is further configured to:

verify, based on the random number generation rule that is recorded in the audit log and corresponds to the random number transmitted by the computing node participating in the secure multi-party computation, whether the random number is a random number constructed based on the private data held by the other computing nodes; and

if yes, determine that the computing parameter transmitted by the computing node contains the risk of privacy leakage;

if no, determine that the computing parameter transmitted by the computing node does not contain the risk of privacy leakage.

13. An electronic device, comprising:

a processor; and

a memory configured to store machine-executable instructions, wherein

by reading and executing the machine-executable instructions that are stored in the memory and correspond to control logic of secure multi-party computation, the processor is caused to perform the method according to any one of claims 1-6.