

the data owner effectively loses control over the data and often has no guarantees that the third-party with whom the data was shared will continue to uphold the privacy of the data.

Keeping personal information private is not trivial, even if an individual chooses not to upload or otherwise share it because of increasing threat of cybercrime. Furthermore, not sharing data at all is highly unlikely and almost impossible as people are regularly required to share personal information when performing transactions via networks and enrolling for services, or share their personal data as part of a legitimate user agreement. The growing concern of sharing private information creates a problem for various entities who operate via online networks. Hence, fundamental questions are raised, such as how individuals can protect their data, and therefore their privacy in the digital realm.

The preceding discussion of the background to the invention is intended only to facilitate an understanding of the present invention. It should be appreciated that the discussion is not an acknowledgment or admission that any of the material referred to was part of the common general knowledge in the art as at the priority date of the application.

SUMMARY OF THE INVENTION

According to an aspect of the present invention there is provided a computer-implemented method for processing anonymized data, carried out at a host computing system including multiple host nodes and comprising: accessing a set of data from multiple data sources each providing a randomized dataset including a true element and at least one spurious element, wherein the host computing system cannot differentiate between the elements so as to obfuscate the true element from the host computing system, wherein a true hash value of the true element is provided to a third-party computing system; wherein each of the host nodes has access to all possible permutations of the randomized datasets and includes compressing the permutations across the host nodes; accessing a code which is executable on the set of data so as to output multiple results for the elements of the set of data; processing the set of data, including for each element: executing the code on the element to generate a result; computing a hash value of the element; and outputting the result in association with the hash value to the third-party computing system, wherein the third-party computing system has access to the true hash value of the true element for identification of the result generated by execution of the code on the true element; and wherein the method includes the host computing system reducing storage requirements of the set of data by executing a number of identical protocol rounds including: creating a random ordering of at least two host nodes with sufficient computing and storing capacity to determine a database host node and at least one aggregator host node; receiving privately at the database host node randomized datasets from a prime number of data sources; the database host node

computing all possible permutations of elements in the received datasets and computing a hash of each permutation and privately sending a hash of a permutation to each data source; receiving, at the aggregator host node, a summed multiparty secret from each data source computed amongst the multiple data sources of a secret of each data source in a finite field over the prime number of data sources together with a hash of the permutation of the data source; and the aggregator host node computing a sum of all the secret shares and storing the sum and the hash of the permutations and communicating a set of hashes of the permutations to the database host node to implement a compression factor as a cut-off to reduce storage requirements for any prime number of data sources.

10

Accessing the set of data may include accessing from each of multiple data sources a randomized dataset including a true element and at least one spurious element; and wherein the host computing system includes multiple host nodes each of which has access to all possible permutations of the randomized datasets and includes compressing the permutations across the host nodes. Each host node may process the set of data by executing a different version of the code. Compressing the permutations across the host nodes may reduce storage requirements of

15

The method may also comprise steps carried out at a data source including: providing a randomized dataset including a true element and at least one spurious element, wherein the spurious element cannot be distinguished from the true element by a host computing system so as to obfuscate the true element from the host computing system; and providing a hash value of the true element to the third-party computing system for later identification of a processed result of the true element.

According to another aspect of the present invention there is provided a system for processing anonymized data comprising a host computing system including multiple host nodes, a processor and a memory configured to provide computer program instructions to the processor to execute functions of components: a data accessing component for accessing a set of data from multiple data sources each providing a randomized dataset including a true element and at least one spurious element, wherein the host computing system cannot differentiate between the elements so as to obfuscate the true element from the host computing system, wherein a true hash value of the true element is provided to the third-party computing system; and wherein each of the host nodes has access to all possible permutations of the randomized datasets and includes compressing the permutations across the host nodes; a code accessing component for accessing a code which is executable on the set of data so as to output multiple results for the elements of the set of data; a code executing component for processing the set of data, including for each element: executing the code on the element to generate a result; computing a hash value of the element; and outputting the result in association with the hash value to the third-party computing system, wherein the third-party computing system has access to the true hash value of the true element for identification of the result generated by execution of the code on the true element; and a storage reduction component for compressing permutations across multiple host nodes included in the system, wherein each of the host nodes has access to all possible permutations of the randomized datasets, and wherein the storage reduction component is configured to reduce storage requirements of the set of data at the host computing system by executing a number of identical protocol rounds including: creating a random ordering of at least two host nodes with sufficient computing and storing capacity to determine a database host node and at least one aggregator host node; receiving privately at the database host node randomized datasets from a prime number of data sources; the database host node computing all possible permutations of elements in the received datasets and computing a hash of each permutation and privately sending a hash of a permutation to each data source; receiving, at the aggregator host node, a summed multiparty secret from each data source computed amongst the multiple data sources of a secret of each data source in a finite field over the prime number of data sources together with a hash of the permutation of the data source; and the aggregator host node computing a sum of all the secret shares and storing the sum and the hash of the permutations and communicating

a set of hashes of the permutations to the database host node to implement a compression factor as a cut-off to reduce storage requirements for any prime number of data sources..

5 The data accessing component may access, from each of multiple data sources, a randomized dataset including a true element and at least one spurious element; and wherein the host computing system includes multiple host nodes each of which has access to all possible permutations of the randomized datasets and including a storage reduction component for compressing the permutations across the host nodes. The storage reduction component may reduce storage requirements of the set of data at the host computing system by applying a secure
10 multiparty computation using secret shares from the multiple data sources.

The system may include a collusion prevention component for preventing collusion between the third-party computing system and the host computing system by means of a tumbling mechanism that obfuscates which host node has computed which result.

15

The system may include an access control component for verifying a license between a data source and a third-party computing system to execute code on the set of data, wherein the data source records a hash of the code in a license associated with the data source, and wherein the

CLAIMS:

1. A computer-implemented method for processing anonymized data, carried out at a host computing system including multiple host nodes and comprising:

5 accessing a set of data from multiple data sources each providing a randomized dataset including a true element and at least one spurious element, wherein the host computing system cannot differentiate between the elements so as to obfuscate the true element from the host computing system, wherein a true hash value of the true element is provided to a third-party computing system;

10 wherein each of the host nodes has access to all possible permutations of the randomized datasets and includes compressing the permutations across the host nodes:

 accessing a code which is executable on the set of data so as to output multiple results for the elements of the set of data;

 processing the set of data, including for each element:

15 executing the code on the element to generate a result;

 computing a hash value of the element; and

 outputting the result in association with the hash value to the third-party computing system, wherein the third-party computing system has access to the true hash value of the true element for identification of the result generated by execution of the code on the true element; and

20 wherein the method includes the host computing system reducing storage requirements of the set of data by executing a number of identical protocol rounds including:

creating a random ordering of at least two host nodes with sufficient computing and storing capacity to determine a database host node and at least one aggregator host node;

25 receiving privately at the database host node randomized datasets from a prime number of data sources;

the database host node computing all possible permutations of elements in the received datasets and computing a hash of each permutation and privately sending a hash of a permutation to each data source;

30 receiving, at the aggregator host node, a summed multiparty secret from each data source computed amongst the multiple data sources of a secret of each data source in a finite field over the prime number of data sources together with a hash of the permutation of the data source; and

35 the aggregator host node computing a sum of all the secret shares and storing the sum and the hash of the permutations and communicating a set of hashes of the permutations to the database host node to implement a compression factor as a cut-off to reduce storage requirements for any prime number of data sources.

2. ~~The method as claimed in claim 1, wherein~~

~~accessing the set of data includes accessing from each of multiple data sources a randomized dataset including a true element and at least one spurious element; and wherein~~

5 ~~the host computing system includes multiple host nodes each of which has access to all possible permutations of the randomized datasets and includes compressing the permutations across the host nodes.~~

10 32. The method as claimed in claim 21, wherein each host node processes the set of data by executing a different version of the code.

4. ~~The method as claimed in claim 2 or claim 3, wherein compressing the permutations across the host nodes reduces storage requirements of the set of data at the host computing system by applying a secure multiparty computation using secret shares from the multiple data sources.~~

5. ~~The method as claimed in claim 4, wherein reducing storage requirements at the host system includes executing a number of identical protocol rounds including:~~

20 ~~creating a random ordering of at least two host computing devices with sufficient computing and storing capacity to determine a database host node and at least one aggregator host node;~~

~~receiving privately at the database host node randomized datasets from a prime number of data sources;~~

25 ~~the database host node computing all possible permutations of elements in the received datasets and computing a hash of each permutation and privately sending a hash of a permutation to each data source;~~

~~receiving at the aggregator host node a summed multiparty secret from each data source computed amongst the multiple data sources of a secret of each data source in a finite field over the prime number of data sources together with a hash of the permutation of the data source; and~~

30 ~~the aggregator host node computing a sum of all the secret shares and storing the sum and the hash of the permutations and communicating a set of hashes of the permutations to the database host to implement a compression factor as a cut-off to reduce storage requirements for any prime number of data sources.~~

35 63. The method as claimed in ~~any one of claims 2 to 5~~claim 1 or 2, including preventing collusion between the third-party computing system and the host computing system by means of a tumbling mechanism that obfuscates which host node has computed which result.

~~74.~~ The method as claimed in any one of the preceding claims, including verifying a license between a data source and a third-party computing system to execute code on the set of data, wherein the data source records a hash of the code in a license associated with the data source, and wherein the code is previously agreed on code if a hash of the executed code, which is stored in a license associated with a code provider, matches the hash of the code stored in the license associated with the data source.

~~85.~~ The method as claimed in any one of the preceding claims, further comprising a method carried out at the third-party computing system including:

receiving from the data source the true hash value of the true element;
receiving from the host computing system the result of code executed on each of the elements of the set of data together with the hash value of each element; and
identifying the result of the true element by matching the true hash value received from the data source to the hash value associated with results from the host computing system.

~~96.~~ The method as claimed in claim ~~85~~, wherein the method carried out at the third-party computing system includes implementing a consensus protocol to verify that the host computing system has executed an agreed code by ensuring that a certain number of host nodes report a same hash of executed code.

~~107.~~ The method as claimed in claim ~~8-5~~ or claim ~~96~~, wherein the method carried out at the third-party computing system includes:

providing access to a true hash value of the true element only where consent to run code on the dataset of the data source is agreed.

~~118.~~ The method as claimed in any one of the preceding claims and further comprising a method carried out at a data source including:

providing a randomized dataset including a true element and at least one spurious element, wherein the spurious element cannot be distinguished from the true element by the host computing system so as to obfuscate the true element from the host computing system; and
providing a hash value of the true element to the third-party computing system for later identification of a processed result of the true element.

~~129.~~ A system for processing anonymized data comprising a host computing system including multiple host nodes, a processor and a memory configured to provide computer program instructions to the processor to execute functions of components:

a data accessing component for accessing a set of data from multiple data sources each providing a randomized dataset including a true element and at least one spurious element,

wherein the host computing system cannot differentiate between the elements so as to obfuscate the true element from the host computing system, wherein a true hash value of the true element is provided to a third-party computing system;

and wherein each of the host nodes has access to all possible permutations of the randomized datasets and includes compressing the permutations across the host nodes;

a code accessing component for accessing a code which is executable on the set of data so as to output multiple results for the elements of the set of data;

a code executing component for processing the set of data, including for each element:

executing the code on the element to generate a result;

computing a hash value of the element; and

outputting the result in association with the hash value to the third-party computing system,

wherein the third-party computing system has access to the true hash value of the true element for identification of the result generated by execution of the code on the true element; and

a storage reduction component for compressing permutations across multiple host nodes included in the system, wherein each of the host nodes has access to all possible permutations of the randomized datasets, and

wherein the storage reduction component is configured to reduce storage requirements of the set of data at the host computing system by executing a number of identical protocol rounds including:

creating a random ordering of at least two host nodes with sufficient computing and storing capacity to determine a database host node and at least one aggregator host node;

receiving privately at the database host node randomized datasets from a prime number of data sources;

the database host node computing all possible permutations of elements in the received datasets and computing a hash of each permutation and privately sending a hash of a permutation to each data source;

receiving, at the aggregator host node, a summed multiparty secret from each data source computed amongst the multiple data sources of a secret of each data source in a finite field over the prime number of data sources together with a hash of the permutation of the data source; and

the aggregator host node computing a sum of all the secret shares and storing the sum and the hash of the permutations and communicating a set of hashes of the permutations to the database host node to implement a compression factor as a cut-off to reduce storage requirements for any prime number of data sources.

~~13. The system as claimed in claim 12, wherein~~

~~the data accessing component accesses from each of multiple data sources a randomized dataset including a true element and at least one spurious element; and wherein~~

~~the host computing system includes multiple host nodes each of which has access to all possible permutations of the randomized datasets and including a storage reduction component for compressing the permutations across the host nodes.~~

~~14. The system as claimed in claim 13, wherein the storage reduction component is for reducing storage requirements of the set of data at the host computing system by applying a secure multiparty computation using secret shares from the multiple data sources.~~

~~15~~ ~~10~~ ~~15~~ ~~10~~. The system as claimed in claim ~~13 or claim 14~~, including a collusion prevention component for preventing collusion between the third-party computing system and the host computing system by means of a tumbling mechanism that obfuscates which host node has computed which result.

~~16~~ ~~11~~. The system as claimed in ~~any one of claims 12 to 15~~ claim 9 or claim 10, including an access control component for verifying a license between a data source and a third-party computing system to execute code on the set of data, wherein the data source records a hash of the code in a license associated with the data source, and wherein the code is previously agreed on code if a hash of the executed code, which is stored in a license associated with a code provider, matches the hash of the code stored in the license associated with the data source.

~~17~~ ~~12~~. The system as claimed in any one of claims ~~12-9~~ to ~~16~~ 11, further comprising a third-party computing system including a processor and a memory configured to provide computer program instructions to the processor to execute functions of components:

a hash receiving component for receiving from a data source a true hash value of the true element;

a result receiving component for receiving from the host computing system a result of code executed on each of the elements of the set of data together with a hash value of each element;

and

a result identifying component for identifying the result of the true element by matching the hash values received from the data source to the hash value associated with results from the host computing system.

~~18~~ ~~13~~. The system as claimed in claim ~~17~~ 12, wherein the third-party computing system includes a consensus protocol component for implementing a consensus protocol to verify that the host computing system has computed an agreed code by ensuring that a certain number of host nodes report a same hash of executed code.

~~19~~14. The system as claimed in in any one of claims ~~12-9~~ to ~~18~~13, further comprising a data source including a processor and a memory configured to provide computer program instructions to the processor to execute functions of components:

5 a dataset providing component for providing a randomized dataset including a true element and at least one spurious element, wherein the spurious element cannot be distinguished from the true element by the host computing system so as to obfuscate the true element from the host computing system; and

10 a hash providing component for providing a hash value of the true element to the third-party computing system for later identification of a processed result of the true element.

~~20. — A computer program product for processing anonymized data, the computer program product comprising a computer-readable medium having stored computer-readable program code for performing the steps of:~~

15 ~~accessing a set of data from a data source including a true element and at least one spurious element, wherein the host computing system cannot differentiate between the elements so as to obfuscate the true element from the host computing system, wherein a true hash value of the true element is provided to a third-party computing system;~~

20 ~~accessing a code which is executable on the set of data so as to output multiple results for the elements of the set of data;~~

~~processing the set of data, including for each element:~~

~~executing the code on the element to generate a result;~~

~~computing a hash value of the element; and~~

25 ~~outputting the result in association with the hash value to the third-party computing system,~~

~~wherein the third-party computing system has access to the true hash value of the true element for identification of the result generated by execution of the code on the true element.~~