INTERNATIONAL STANDARD

ISO/IEC 4922-1

First edition
2023-07

Information security — Secure multiparty computation —

Part 1: General

Reference number
ISO/IEC 4922-1:2023(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword ..... **iv**
Introduction ..... **v**
1 Scope **1**
2 Normative references **1**
3 Terms and definitions **1**
4 General model and parameters **2**
4.1 Generic model 2
4.2 Parameters of secure multiparty computation 4
4.2.1 Overview 4
4.2.2 Input space 4
4.2.3 Encoded space 4
4.2.4 Output space 4
4.2.5 The number of computing parties 4
4.2.6 Role restriction 4
4.2.7 Communication model 4
4.2.8 Summary of parameters 5
5 Properties and analysis of secure multiparty computation **5**
5.1 Fundamental requirements 5
5.1.1 Overview 5
5.1.2 Correctness 5
5.1.3 Input privacy 5
5.2 Adversary model 5
5.2.1 Overview 5
5.2.2 Adversary behaviour 6
5.2.3 Number of corruptions 6
5.2.4 Computational power 6
5.2.5 Composition and parallel execution 7
5.2.6 Network access 7
5.3 Optional properties 7
5.3.1 Overview 7
5.3.2 Correctness against active adversary 7
5.3.3 Input privacy against active adversary 7
5.3.4 Fairness 8
5.3.5 Guaranteed output delivery 8
5.4 Performance properties for the comparison of schemes 8
5.4.1 Overview 8
5.4.2 Communication efficiency 8
5.4.3 Computational efficiency 8
Annex A (informative) **Possible use cases for secure multiparty computation** **9**
Bibliography **10**

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *Information security, cybersecurity and privacy protection*.

A list of all parts in the ISO/IEC 4922 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

Secure multiparty computation (MPC) is a cryptographic technique that enables the output of a function to be computed while keeping the individual inputs, provided by a range of parties, secret. It is a valuable tool to improve privacy in situations where computations are outsourced, or where different distrusting stakeholders are required to cooperate, and no trusted party is available to execute the computation on behalf of the input providers.

Secure multiparty computation is a decentralized protocol which emulates the functionality of a trusted third party, taking the private inputs of individual players, computing an agreed function, and disseminating the correct output privately to relevant parties.

Secure multiparty computation is useful in situations where mutually distrusting entities want to collaborate on data processing tasks, which can arise in the Internet of Things and other distributed application domains. Possible application domains include secure auctions, privacy-preserving data analytics, and distributed digital wallets.

Annex A provides possible use cases for secure multiparty computation.

Information security — Secure multiparty computation —

Part 1: General

1 Scope

This document specifies definitions, terminology and processes for secure multiparty computation and related technology, in order to establish a taxonomy and enable interoperability. In particular, this document defines the processes involved in cryptographic mechanisms which compute a function on data while the data are kept private; the participating parties; and the cryptographic properties.

The terminology contained in this document is common to the ISO/IEC 4922 series.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

3.1
intended function
function to be evaluated by the *multiparty protocol* (3.2)

3.2
multiparty protocol
protocol executed among *computing parties* (3.6) to jointly evaluate the *intended function* (3.1) over encoded inputs

3.3
input
private data held by the *input party* (3.5) for the purpose of being evaluated by the *intended function* (3.1)

3.4
party
entity involved in secure multiparty computation

3.5
input party
party (3.4) holding an input and providing it to *computing parties* (3.6) in encoded form

3.6
computing party
party (3.4) that performs computations to evaluate the *intended function* (3.1)

3.7
result party
party (3.4) receiving the required data from the *computing parties* (3.6) to obtain the result of the secure multiparty computation in plaintext by decoding the output of the *intended function* (3.1)

3.8
circuit
representation of the *intended function* (3.1) in the form of basic operations supported by the protocol and their interconnections

3.9
arithmetic circuit
circuit (3.8) composed of basic arithmetic operations

3.10
boolean circuit
circuit (3.8) composed of basic Boolean operations

3.11
communication complexity
total amount of data transferred among the *computing parties* (3.6) during the execution of a *multiparty protocol* (3.2)

3.12
computational complexity
amount of computation required to execute a *multiparty protocol* (3.2)

3.13
round complexity
minimum number of sequential communications between *computing parties* (3.6) required during the execution of a *multiparty protocol* (3.2)

4 General model and parameters

4.1 Generic model

In a secure multiparty computation, two or more parties are involved in a protocol to evaluate an intended function on private inputs. A party learns nothing about the inputs of other parties except what it can deduce from the output and its own input. This security property can be useful in various application scenarios. Annex A provides possible use cases for secure multiparty computation.

Different roles shall be present in a secure multiparty computation system. The roles are:

— input party,

— computing party,

— result party.

The input parties hold the inputs for the intended function to be evaluated in secure multiparty computation. Each input party encodes its input and distributes the encodings to the computing parties. The encoding ensures that the input is kept private from the other parties and can be achieved by secret sharing or encryption.

NOTE For secret sharing, see the ISO/IEC 19592 series. For encryption, see the ISO/IEC 18033 series.

The computing parties jointly execute the multiparty protocol steps necessary to evaluate the intended function and disseminate the derived output encodings to the result parties.

The result parties reconstruct the result of the computation from the encoded outputs. One or more result parties can be present, depending on the use case. For example, in the case of a secure auction, all input parties are also interested in the result and likely to be result parties. Different result parties can also receive different results as the output of different functions, if incorporated in the intended function.

In a secure multiparty computation system, all roles shall be present and there can be multiple parties serving in each role. Each party can occupy any number of roles. Specifically, to qualify for a secure multiparty computation system, at least two computing parties shall be present to evaluate the intended function.

EXAMPLE The millionaires' problem,[13] determining who is wealthier without revealing their actual wealth, is a classic problem within secure multiparty computation. The intended function evaluated by the computing parties is a comparison function. In a conventional two-party protocol solving the millionaires' problem, each party provides an input and contributes to the evaluation of the comparison function, but only one party gets the final output. Therefore, both parties are input parties, and both are also computing parties. But only one of them is a result party.

The processing of an intended function is divided into simple operations which are represented as primitives. The primitives, such as types of gates, supported by a secure multiparty computation depend on the available protocols. The function is computed by composing those primitives. In the case of arithmetic circuits, it is composed of elements such as addition, subtraction, scalar-multiplication, and multiplication gates. For Boolean circuits, it typically consists of logic operations on binary values; more complex gates are also possible.

A secure multiparty computation comprises the following steps.

a) An intended function shall be agreed upon among the involved parties. The parties involved in this agreement depend on the application. In a typical case, this can be all the input parties.

b) The input parties generate encodings of their input and distribute them to the computing parties.

c) The computing parties evaluate the intended function over their encoded inputs according to predefined rules for the specified protocol producing the encoded result. During the computation, the computing parties can communicate with each other for certain protocol steps.

d) The encoding of the result of the computation held by the computing parties is sent to the corresponding result parties which can decode it into the plaintext.

Figure 1 illustrates an example of secure multiparty computation with i input parties (I_1, I_2, ..., I_i), m computing parties (C_1, C_2, ..., C_m), and n result parties (R_1, R_2, ..., R_n).

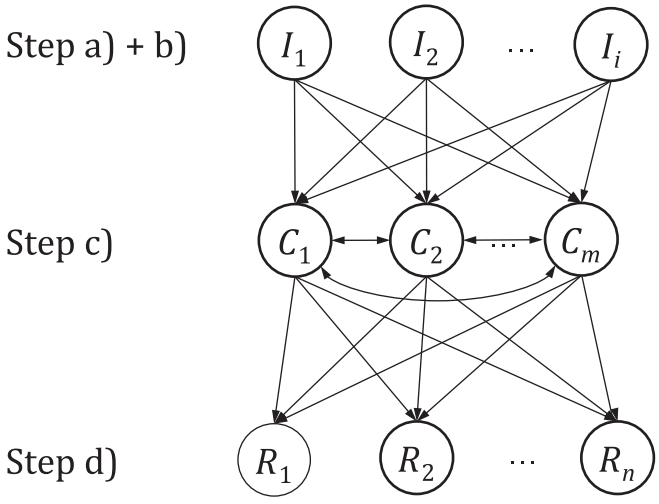

Figure 1 — Example of secure multiparty computation with input, computing, and result parties

When designing protocols, it is important to take into account that secure multiparty computation does not necessarily protect against a party deducing potentially private information from the output of the computation and its own input. For example, to compute the mean of two values, each input party that is also a result party can compute the other party's input using the mean and its own input.

4.2 Parameters of secure multiparty computation

4.2.1 Overview

The following basic set of parameters apply to all secure multiparty computation schemes specified in the ISO/IEC 4922 series; they provide a means to evaluate and compare the properties of particular protocols:

— the input space, described in 4.2.2;

— the encoded space, described in 4.2.3;

— the output space, described in 4.2.4;

— the number of computing parties, described in 4.2.5;

— the role restriction, described in 4.2.6;

— the communication model, described in 4.2.7.

4.2.2 Input space

The input space is the set of possible values the input parties can have.

4.2.3 Encoded space

The encoded space is the set of possible values for the encoded values operated on by the computing parties.

4.2.4 Output space

The output space is the set of possible values output by the intended function. In other words, these are the possible values that can be reconstructed by the result parties as a result of the multiparty protocol.

4.2.5 The number of computing parties

The number of computing parties executing the multiparty protocol shall be at least two.

4.2.6 Role restriction

A role restriction is a description of any restriction on the roles which specific parties may perform in a secure multiparty computation. For example, all the input parties can also be the computing parties.

4.2.7 Communication model

4.2.7.1 Overview

A communication model expresses how to exchange data among parties. Secure multiparty computation needs access to at least one of the following two options described in 4.2.7.2 and 4.2.7.3, but sometimes also both.

4.2.7.2 Point-to-point channel

This communication model consists of a fully connected network with a point-to-point communication channel between each pair of parties. This is the typical setting for secure multiparty computation.

4.2.7.3 Broadcast and multicast channel

A broadcast channel is a special communication channel that guarantees the correct delivery of transmitted messages from a sender to all parties. That is, all honest parties eventually receive the same transmitted message. If the channel delivers the message only to a subset of the parties, it is called multicast channel.

NOTE Broadcast channels can be realized over point-to-point channels.

4.2.8 Summary of parameters

To illustrate the parameters, the example from 4.1 is continued.

EXAMPLE In the protocol given in Reference [9] for solving the two-party millionaires' problem, the input space is the set of positive integers less than 2^d, where d is the bit length of the input space. The two parties are both input parties and also computing parties; only one of them is a result party under the role restriction. Therefore, in this example, there is no explicit encoded space. The output space is a single bit indicating the result of the comparison. Under this simple two-party protocol, the communication models of point-to-point and broadcast/multicast are equivalent.

5 Properties and analysis of secure multiparty computation

5.1 Fundamental requirements

5.1.1 Overview

The two fundamental requirements met by secure multiparty computation in the ISO/IEC 4922 series are correctness and input privacy. The correctness requirement demands that the output is computed according to the given function and its inputs. The input privacy requirement demands that the secrecy of the input messages is maintained.

5.1.2 Correctness

If all the parties follow the protocol, the result parties shall obtain the output of an intended function over the inputs provided by the input parties.

5.1.3 Input privacy

If all the parties follow the protocol and no adversary corrupts more than the adversarial model in 5.2.3 allows, it shall not be possible for any party to obtain additional information about inputs, except what the party may know by its role. For example, an input party which is also a result party knows its input and output, respectively. Input privacy is also sometimes referred to as input secrecy or input security.

NOTE The information required to compute a function, such as its topology and the number of inputs, is assumed to be available to all parties.

5.2 Adversary model

5.2.1 Overview

The security of secure multiparty computation is determined by the type of adversary the protocol is secure against. 5.2 describes models that classify an adversary according to its capabilities.

5.2.2 Adversary behaviour

5.2.2.1 Passive security model

In the passive security model, an adversary follows the protocol honestly, yet can try to learn additional information from the available information. Passive security is also known as semi-honest or honest-but-curious security.

5.2.2.2 Active security model

In the active security model, an adversary can arbitrarily deviate from the protocol in their attempt to learn more information about the other parties' data or to produce an incorrect result. Active security is also known as malicious security.

5.2.2.3 Covert security model

If the cheating behaviours of malicious adversaries have a probability λ of being caught by the honest parties, a secure multiparty computation scheme satisfies the covert security model with a deterrence factor λ. The covert security model is a trade-off between the passive and active security models. Additionally, if the honest party can generate a publicly verifiable proof of the adversary's cheating behaviour without sacrificing the input privacy, a secure multiparty computation scheme satisfies the publicly verifiable covert security model.

5.2.3 Number of corruptions

5.2.3.1 General

An adversary can corrupt one or more of the input, computing, or result parties. Secure computation mechanisms can be categorized by how many corruptions the mechanisms can tolerate, called the threshold. Typical examples include, but are not limited to, 5.2.3.2 and 5.2.3.3.

5.2.3.2 Honest majority

In the honest majority case, an adversary can corrupt less than half of the computing parties.

5.2.3.3 Dishonest majority

In the dishonest majority case, an adversary can corrupt half or more of the computing parties.

5.2.4 Computational power

5.2.4.1 General

An adversary can be modelled with different computing capabilities. Secure computation mechanisms can be categorized by the adversarial computing power they can resist. Typical examples include, but are not limited to, 5.2.4.2 and 5.2.4.3.

5.2.4.2 Information-theoretic security

For information-theoretic security, an adversary can have unbounded computational power.

5.2.4.3 Computational security

For computational security, an adversary has only polynomial-time computational power.

EXAMPLE The protocol for solving the two-party millionaires' problem defined in Reference [9] has computational security but not information-theoretical security.

5.2.5 Composition and parallel execution

An adversary can have the capability to execute multiple secure multiparty computation instances sequentially, in parallel, or concurrently. Protocols secure under composition can cope with such adversaries.

5.2.6 Network access

5.2.6.1 General

An adversary can have different levels of access to the communication network. Secure computation mechanisms can be categorized by the channel model they rely on. Typical examples include, but are not limited to, 5.2.6.2 to 5.2.6.4.

5.2.6.2 Unprotected channels

In the unprotected channels case, an adversary has full access to the channels which do not provide any security per se, meaning an adversary can tamper with transmitted messages.

5.2.6.3 Authenticated channel

The authenticated channel model guarantees the authenticity of the transmitted messages. An adversary can read all transmitted messages and delay, forward or delete them, but cannot inject additional messages.

5.2.6.4 Secure channel

The secure channel communication model guarantees the confidentiality and authenticity of transmitted messages. An adversary can only forward (with delay) or delete transmitted messages.

NOTE The adversary can learn general information about the transmitted messages, such as the message length.

EXAMPLE The protocol for solving the two-party millionaires' problem defined in Reference [9] requires an authenticated channel between the two parties.

5.3 Optional properties

5.3.1 Overview

Apart from the fundamental requirements, secure multiparty computation can exhibit other properties relating to its parameters and processes.

5.3.2 Correctness against active adversary

In the correctness against malicious behaviour case, correctness as defined in 5.1.2 holds, even if the corrupted parties are active, that is, they deviate from the protocol. This correctness goal comes in two variants: either the honest result parties are guaranteed to obtain the correct output (a robust protocol with fairness or guaranteed output delivery), or they abort if one of them will receive an incorrect output (a secure multiparty computation protocol with abort or covert security).

5.3.3 Input privacy against active adversary

In the input privacy against malicious behaviour case, input privacy as defined in 5.1.3 holds, even if the corrupted parties are active, which means they deviate from the protocol.

5.3.4 Fairness

A scheme provides fairness if corrupted result parties shall receive their output, only if the honest result parties also receive their outputs.

5.3.5 Guaranteed output delivery

Guaranteed output delivery means that honest result parties always obtain their output whatever the corrupted parties do. This means corrupted parties cannot prevent honest result parties from receiving their output.

NOTE A multiparty protocol with the guaranteed output delivery property will also have the fairness property, but the opposite is not true.

5.4 Performance properties for the comparison of schemes

5.4.1 Overview

The communication and computational efficiency of a scheme depend on the number of parties, the size of the inputs, and/or the function to be evaluated. Descriptions of efficiency can be given in concrete or asymptotic terms.

5.4.2 Communication efficiency

The communication efficiency of a multiparty protocol is given by round complexity and communication complexity.

EXAMPLE The protocol for solving the millionaires' problem defined in Reference [9] has two rounds with a total communication complexity $O(d^2)$, where the input space is the set of positive integers less than 2^d and $O(.)$ is the notation used to describe asymptotic function behaviour.

5.4.3 Computational efficiency

The computational efficiency of a multiparty protocol is given by the computational complexity.

EXAMPLE The protocol for solving the millionaires' problem defined in Reference [9] has computational complexity $O(d^2)$, where the input space is the set of positive integers less than 2^d and $O(.)$ is the notation used to describe asymptotic function behaviour.