INTERNATIONAL STANDARD

ISO/IEC 19592-2

First edition
2017-10

Information technology — Security techniques — Secret sharing —

Part 2:
Fundamental mechanisms

Technologies de l'information — Techniques de sécurité — Partage de secret —

Partie 2: Mécanismes fondamentaux

Reference number
ISO/IEC 19592-2:2017(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

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. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

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).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

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

For an explanation on 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 the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT security techniques*.

A list of all parts in the ISO/IEC 19592 series can be found on the ISO website.

Introduction

A secret sharing scheme is a cryptographic technique used to protect the confidentiality of a message by dividing it into a number of pieces called shares. A secret sharing scheme has two main parts: a message sharing algorithm for dividing the message into shares and a message reconstruction algorithm for recovering the message from all or a subset of the shares.

The fundamental functions of a secret sharing scheme are sharing and reconstructing the message. A secret sharing scheme can also have optional features such as reconstructing the message when some shares provided for reconstruction are erroneous. This document specifies cryptographic secret sharing schemes which possess the two fundamental functions of message confidentiality and message recoverability.

Secret sharing can be used to store data (for example, confidential values or cryptographic keys) securely in distributed systems. Moreover, secret sharing is a fundamental technology for secure multi-party computation that can be used to protect the processing of data in a distributed system. To facilitate the effective use of the technology and to maintain interoperability, ISO/IEC 19592 (all parts) specifies secret sharing and related technology.

NOTE Annex A lists the object identifiers assigned to the secret sharing fundamental mechanisms specified in this document. Annex B provides numerical examples.

Information technology — Security techniques — Secret sharing —

Part 2: Fundamental mechanisms

1 Scope

This document specifies cryptographic secret sharing schemes.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 19592-1:2016, *Information technology — Security techniques — Secret sharing — Part 1: General*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 19592-1 and the following apply.

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

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

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

3.1
abelian group
group (3.8) (G, +) such that $a + b = b + a$ for every a and b in G

[SOURCE: ISO/IEC 15946-1:2016, 3.1, modified]

3.2
complexity
number of unit operations required to execute a procedure

3.3
conversion protocol
protocol that converts the shares of a secret sharing scheme to the shares of another secret sharing scheme

3.4
deterministic random bit generator
DRBG
random bit generator that produces a random-appearing sequence of bits by applying a deterministic algorithm to a suitably random initial value called a seed and, possibly, some secondary inputs upon which the security of the random bit generator does not depend

Note 1 to entry: A DRBG takes a high-entropy, kept-secret random string as input and outputs a longer string of bits which is computationally indistinguishable from random data to adversaries not knowing the input.

[SOURCE: ISO/IEC 18031:2011, 3.10, modified]

3.5
field
set of elements K and a pair of operations (+, *) defined on K such that: (i) $a * (b + c) = a * b + a * c$ for every a, b and c in K, (ii) K together with + forms an *abelian group* (3.1) (with identity element 0), and (iii) K excluding 0 together with * forms an abelian group

[SOURCE: ISO/IEC 15946-1:2016, 3.4, modified]

3.6
finite cyclic group
abelian group (3.1) G such that there exist g in G, where every a in G is specified in g or a self-addition of g

3.7
finite field
field (3.5) containing a finite number of elements

[SOURCE: ISO/IEC 15946-1:2016, 3.5, modified]

3.8
group
set of elements G and an operation + defined on the set of elements such that (i) $a + (b + c) = (a + b) + c$ for every a, b and c in G, (ii) there exists an identity element e in G such that $a + e = e + a = a$ for every a in G, and (iii) for every a in G there exists an inverse element a^{-1} in G such that $a + a^{-1} = a^{-1} + a = e$

[SOURCE: ISO/IEC 15946-1:2016, 3.6, modified]

3.9
information dispersal algorithm
IDA
algorithm that includes two separated sub-algorithms: a splitting algorithm that splits a message into n components and a recover algorithm that recovers the message from any k of the n components, where k and n are integers and $n \geq k$

Note 1 to entry: Unlike in a secret sharing scheme, there is no guarantee of security. That is, it can be possible to reconstruct the secret or parts of the secret from less than k components.

4 Symbols and abbreviated terms

$a \in A$	a is an element of A
$A \subset B$	A is a subset of B
$\|A\|$	number of elements of A
$A \times B$	direct product of A and B
A^m	set of m-tuples of elements of A
${}_i\mathrm{C}_j$	binomial coefficient, namely i choose j
$[a]_i$	i-th share of secret a
n	number of shares
k	threshold of shares
G	finite cyclic group
K	finite field

$K[x]$	set of all polynomials in x with coefficient in K
Split	message splitting algorithm of an IDA scheme
Rec	message reconstruction algorithm of an IDA scheme
Share	message sharing algorithm of a secret sharing scheme
Reconst	message reconstruction algorithm of a secret sharing scheme
HomShare	message sharing algorithm of a homomorphic secret sharing scheme
HomReconst	message reconstruction algorithm of a homomorphic secret sharing scheme

5 Secret sharing schemes

5.1 General

In this document, each of 5.2, 5.3, 5.4, 5.5 and 5.6 contains a specification of one or more secret sharing schemes. For each secret sharing scheme, the following items are listed.

a) Parameters

1) Message space, i.e. the set of possible messages which can be input to the message sharing algorithm.

2) Share space, i.e. the set of possible shares which can be output by the message sharing algorithm.

3) Number of shares, i.e. the range of possible values of n supported by the scheme.

4) One of the following properties that represent which shares are required for the reconstruction:

i) Threshold, i.e. a positive number k such that any k shares are sufficient for a successful completion of the message reconstruction algorithm.

ii) Access structure, i.e. the minimal set of possible subsets of shares that are needed as input in order for the message reconstruction algorithm to successfully output the message.

iii) Adversary structure, i.e. the set of subsets of shares that is not possible to reconstruct the message.

5) Other parameters (if applicable).

b) Description of the message sharing algorithm, i.e. the method for dividing a message into shares.

c) Description of the message reconstruction algorithm, i.e. the method for reconstructing the message from a set of shares.

d) Properties of the secret sharing scheme (see ISO/IEC 19592-1:2016, Clause 4).

NOTE 1 None of the secret sharing schemes specified in this document possesses the verifiability property.

NOTE 2 In the mechanisms specified in this document, elements are chosen at random from some (finite) set. All such choices are made uniformly (or near uniformly) at random from the set of possible values.

NOTE 3 If the message space is a group or field, arithmetic operations are performed in this group or field.

5.2 Shamir secret sharing scheme

5.2.1 General

The parameters, message sharing algorithm, message reconstruction algorithm and properties of the Shamir secret sharing scheme[8] are described in 5.2.

5.2.2 Parameters

Message space: K.

Share space: same as the message space.

Number of shares: n, such that $n \geq 2$, $n < |K|$.

Threshold: k, such that $n \geq k \geq 2$.

Fixed field elements: $x_i \in K$ for $1 \leq i \leq n$.

NOTE It is assumed that the fixed field elements are known to the receiver. These elements can be sent to the receiver with the corresponding share or published as system parameters.

5.2.3 Message sharing algorithm

Input: message $a \in K$.

Output: share vector $([a]_1,..., [a]_n) \in K^n$.

a) Randomly select $r_1,..., r_{k-1} \in K$.

b) Compute $[a]_i = a + \sum_{j=1}^{k-1} r_j x_i^j \in K$ for $1 \leq i \leq n$.

c) Output $([a]_1,..., [a]_n) \in K^n$.

5.2.4 Message reconstruction algorithm

Input: share vector $\left([a]_{i_1},...,[a]_{i_k}\right) \in K^k$.

Output: message $a \in K$.

a) Compute $a = \sum_{j=1}^{k} [a]_{i_j} \prod_{u=1,u\neq j}^{k} \left(0 - x_{i_u}\right) / \left(x_{i_j} - x_{i_u}\right) \in K$.

b) Output $a \in K$.

NOTE The reconstruction algorithm is known as Lagrange interpolation. If $f(x) = a + \sum_{j=1}^{k-1} r_j x^j$ then the secret is $f(0)$ and each share $[a]_i$ is $f(x_i)$. Since $f(x)$ is a polynomial of degree k, $f(0)$ can be computed from k coordinates using Lagrange interpolation.

5.2.5 Properties

Confidentiality: The Shamir secret sharing scheme is perfectly information-theoretically confidential when the receiver has access to less than k shares of the message.

Information rate: The Shamir secret sharing has an information rate of 1, as the size of a message and a share are the same as the size of an element of the finite field K. Thus, the scheme is ideal.

Homomorphic operations: The Shamir secret sharing scheme is (+, +)-homomorphic where addition on share vectors is performed by computing $[a + a']_i = [a]_i + [a']_i$.

Complexity: The message sharing algorithm requires $(k–1)n$ multiplications and $(k–1)n$ additions. The message reconstruction algorithm requires k divisions, $2k^2 - 3k$ multiplications and $k^2 - 1$ additions. If anything that does not involve a or r_j for $1 \le j \le k–1$ is preliminary prepared, both algorithms require k multiplications and $k–1$ additions.

5.3 Ramp Shamir secret sharing scheme

5.3.1 General

The parameters, message sharing algorithm, message reconstruction algorithm and properties of the ramp version of the Shamir secret sharing scheme[3] are described in 5.3. This mechanism is a generalization of the scheme specified in 5.2. It reduces the size of each share in relation to the message to be reconstructed by a factor of L. Although k shares are still required to reconstruct the message, any number of shares greater than $(k–L)$ reveals partial information about it. The parameters k and L can be chosen flexibly following the restriction $k \ge L \ge 1$.

NOTE 1 The ramp Shamir secret sharing scheme with the parameter $L = 1$ is equivalent to the Shamir secret sharing scheme specified in 5.2.

NOTE 2 In information-theoretically secure secret sharing schemes, each share of a secret is at least the size of the secret. There are two approaches to mitigate this. One is to rely on computational hardness assumptions instead of information theoretic security. The other is the use of ramp secret sharing schemes. In the ramp scheme, shares can be shorter than the size of the secret, while there are sets of shares that are not meant to allow access but which leak information about the secret.

5.3.2 Parameters

Message space: K^L.

Share space: finite field K.

Number of shares: n, satisfying $n \ge 2$, $n < |K|$.

Threshold: k, satisfying $n \ge k \ge 2$.

Number of embedded messages: L, satisfying $k \ge L \ge 1$.

Fixed field elements: $x_i \in K$ for $1 \le i \le n$.

NOTE The fixed field elements can be sent to the receiver with the corresponding shares or published as system parameters.

5.3.3 Message sharing algorithm

Input: message $(a_1,\dots, a_L) \in K^L$.

Output: share vector $([(a_1,\dots, a_L)]_1,\dots, [(a_1,\dots, a_L)]_n) \in K^n$.

a) Randomly select $r_L,\dots, r_{k-1} \in K$.

b) Compute $\left[\left(a_1,\dots,a_L\right)\right]_i = \sum_{j=0}^{L-1} a_{j+1} x_i^j + \sum_{j=L}^{k-1} r_j x_i^j \in K$ for $1 \le i \le n$.

c) Output $([(a_1,\dots, a_L)]_1,\dots, [(a_1,\dots, a_L)]_n) \in K^n$.

5.3.4 Message reconstruction algorithm

Input: share vector $\left(\left[\left(a_1,\dots,a_L\right)\right]_{i_1},\dots,\left[\left(a_1,\dots,a_L\right)\right]_{i_k}\right)\in K^k$.

Output: messages $(a_1,\dots, a_L) \in K^L$.

a) Define polynomial $f(x)=\sum_{j=1}^{k}\left[\left(a_1,\dots,a_L\right)\right]_{i_j}\prod_{u=1,u\neq j}^{k}\left(x-x_{i_u}\right)/\left(x_{i_j}-x_{i_u}\right)\in K[x]$.

b) Compute $f(x)=\sum_{i=0}^{k-1}b_{i+1}x^i\in K[x]$ and set $a_i = b_i$ for $1 \le i \le L$.

c) Output $(a_1,\dots, a_L) \in K^L$.

5.3.5 Properties

Confidentiality: The ramp version of the Shamir secret sharing scheme is information-theoretically confidential when the receiver has access to less than $k - L + 1$ shares. When the receiver has access to more than $k - L$ shares but less than k, partial information is revealed. This reduction in security is quantified as follows: if an entity knows $k - L + i$ shares for some i $(1 \le i \le L - 1)$ then the entity knows the secret message lies within a set of size $|K|^{L-i}$.

Information rate: The ramp version of the Shamir secret sharing scheme has an information rate of L as the size of a message is equal to L times the size of a field element and the size of a share is the same as the size of a field element.

Homomorphic operations: The ramp version of the Shamir secret sharing scheme is (+, +)-homomorphic where addition on share vectors is performed by computing $[(a_1 + a'_1,\dots, a_L + a'_L)]_i = [(a_1,\dots, a_L)]_i + [(a'_1,\dots, a'_L)]_i$.

Complexity: The message sharing algorithm requires $(k-1)n$ multiplications and $(k-1)n$ additions. The message reconstruction algorithm requires $k(k-1)(k+1)/3$ divisions, $k(k-1)/2$ multiplications and $k(k-1)(2k+5)/6$ additions using the Gaussian elimination method. If anything that does not involve a or r_j for $L \le j \le k-1$ is preliminary prepared, the message sharing algorithm requires k multiplications and $k-1$ additions.

5.4 Additive secret sharing scheme for a general adversary structure

5.4.1 General

The parameters, message sharing algorithm, message reconstruction algorithm and properties of the additive secret sharing scheme for a general adversary structure[5] are described in 5.4. Let A be the adversary structure that contains m subsets of the numbers $\{1, 2,\dots, n\}$ of variable size representing groups of adversaries. The algorithms are arranged so that no set of adversaries in A can collaborate to recover a. Let the elements of A be labelled Z_j, $j = 1, 2,\dots, m$. In the message sharing algorithm, values $r_{Z_1},\dots,r_{Z_{m-1}}$ are generated uniformly at random within the field and $r_{Z_m}=a-\left(r_{Z_1}+\cdots+r_{Z_{m-1}}\right)$. Share $[a]_i$ then consists of all the r values whose indices do not contain the value i, where $i = 1, 2,\dots, n$.

NOTE 1 The adversary structure denotes the set of all maximal coalitions of participants who cannot recover the secret.

NOTE 2 A complementary concept to the notion of the adversary structure of a secret sharing scheme is the access structure of the scheme. The access structure contains all minimal coalitions of participants of the scheme who can jointly recover the secret.

5.4.2 Parameters

Message space: G.

Share space: same as the message space.

Number of shares: n, such that $n \geq 2$.

Adversary structure: $\boldsymbol{A} \subset \{S| \ S \subset\{1,..., n\}\}$.

Fixed subset: $Z_0 \in \boldsymbol{A}$.

NOTE The index $Z \in \boldsymbol{A}$ of r_Z can be sent to the receiver with the corresponding share or be published as system parameter.

5.4.3 Message sharing algorithm

Input: message $a \in G$.

Output: share vector $([a]_1,..., [a]_n)$.

a) Randomly select $r_Z \in G$ for all $Z \in \boldsymbol{A}-\{Z_0\}$ and compute $r_{Z_0} = a - \sum_{Z\in \boldsymbol{A}-\{Z_0\}} r_Z \in G$.

b) Compute $[a]_i = \{r_Z \mid i \notin Z \in \boldsymbol{A}\}$ for $1 \leq i \leq n$.

c) Output $([a]_1,..., [a]_n)$.

5.4.4 Message reconstruction algorithm

Input: share vector $\{[a]_i \mid i \in K\}$, where K satisfies the requirement that for all $Z \in \boldsymbol{A}$, there exists $i_Z \in K$ such that $i_Z \notin Z$.

Output: message $a \in G$.

a) Extract $r_Z \in G$ from share $[a]_{i_Z}$ for all $Z \in \boldsymbol{A}$.

b) Compute $a = \sum_{Z\in \boldsymbol{A}} r_Z \in G$.

c) Output $a \in G$.

5.4.5 Properties

Confidentiality: The additive secret sharing scheme for a general adversary structure is perfectly information-theoretically confidential when the receiver only has access to shares $\{[a]_i \mid i \in Z\}$ for some $Z \in \boldsymbol{A}$.

Information rate: The information rate for the additive secret sharing scheme for a general adversary structure is $1 / \max_{1\leq i\leq n} \left|\{r_Z \mid i \notin Z \in \boldsymbol{A}\}\right|$, as the size of a message is the same as the element size in G and the size of a share is at most $\max_{1\leq i\leq n} \left|\{r_Z \mid i \notin Z \in \boldsymbol{A}\}\right|$ times the element size. If $|\boldsymbol{A}| = 1$, the scheme is ideal.

Homomorphic operations: The additive secret sharing scheme for a general adversary structure is $(+,+)$-homomorphic where the addition on share vectors is performed by computing $[a]_i + [a']_i = \{r_Z + r'_Z \mid i \notin Z \in \boldsymbol{A}\}$.

Complexity: The message sharing algorithm requires $|\boldsymbol{A}|-1$ additions. The message reconstruction algorithm requires $|\boldsymbol{A}|-1$ additions.

5.5 Replicated additive secret sharing scheme

5.5.1 General

The parameters, message sharing algorithm, message reconstruction algorithm and properties of the replicated additive secret sharing scheme[4] are described in 5.5. In this scheme, each share is considerably larger than the message being shared. However, reconstruction is computationally trivial, depending on the group and the threshold number of shares used as parameters. This scheme is a special case of the secret sharing scheme described in 5.4 with specific adversary structures $\boldsymbol{A} = \{Z \mid Z \subset \{1,..., n\}, |Z| = k-1\}$.

Each party has random numbers corresponding to those adversary sets that do not include the party. Thus, parties that form a subset of a set of size k–1 cannot reconstruct the secret because they do not have the random number corresponding to that set. On the other hand, parties that are not a subset of any set of size k–1 can reconstruct the secret because for any set of size k–1 there exists a party that is not included in that set and that party has the random number corresponding to that set of size k–1.

5.5.2 Parameters

Message space: G.

Share space: same as the message space.

Number of shares: n, such that $n \geq 2$.

Threshold: k, such that $n \geq k \geq 2$.

Adversary structure: $\boldsymbol{A} = \{Z \mid Z \subset \{1,..., n\}, |Z| = k-1\}$.

Fixed subset: $Z_0 \in \boldsymbol{A}$.

NOTE The index $Z \in \boldsymbol{A}$ of r_Z can be sent to the receiver with the corresponding share or published as a system parameter.

5.5.3 Message sharing algorithm

Same as the message sharing algorithm in 5.4.3.

5.5.4 Message reconstruction algorithm

Same as the message reconstruction algorithm in 5.4.4.

5.5.5 Properties

Confidentiality: The replicated additive secret sharing scheme is perfectly information-theoretically confidential when the receiver has access to less than k shares.

Information rate: The information rate for the replicated additive secret sharing scheme is $1/_{n-1}C_{k-1}$. This is because the size of a message is the same as the element size in G and the size of a share is $_{n-1}C_{k-1}$ times the element size.

Homomorphic operations: The replicated additive secret sharing scheme is (+, +)-homomorphic where addition on share vectors is performed by computing $[a]_i + [a']_i = \{r_Z + r'_Z \mid i \notin Z \in \boldsymbol{A}\}$.

Complexity: The sharing algorithm requires $_nC_{k-1}$ –1 additions. The message reconstruction algorithm requires $_nC_{k-1}$ –1 additions.

5.6 Computational additive secret sharing scheme

5.6.1 General

5.6 describes the parameters, message sharing algorithm, message reconstruction algorithm, conversion protocol and properties of the computational additive secret sharing scheme[6][7]. The computational additive secret sharing scheme achieves a large information rate by providing computational confidentiality and discarding homomorphic operations. Homomorphic operations can be performed on the share vectors if they are first converted to a homomorphic secret sharing scheme. This scheme provides such a conversion protocol.

5.6.2 Parameters

Number of shares: n, such that $n \geq 2$.

Threshold: k, such that $n \geq k \geq 2$.

Secret sharing scheme: a message sharing algorithm $\mathrm{Share}:X \rightarrow S^n$, message reconstruction algorithm $\mathrm{Reconst}:S^k \rightarrow X$ that has space X of message, space S of share, the number of shares n and threshold k.

Message space: G.

Seed space: group X.

Number of seeds: $m \geq 1$.

Deterministic random bit generator: $\mathrm{DRBG}:X \rightarrow G$ that takes a seed as input and outputs a pseudo-random element in G.

NOTE 1 Both the input and output of an ordinary DRBG are bit strings, but a DRBG with group element output can be constructed following guidance in ISO/IEC 18031:2011, B.1.

Information dispersal algorithm: IDA consists of $\mathrm{Split}:G \rightarrow N^n$ and $\mathrm{Rec}:N^k \rightarrow G$ that has space G of message, space N of output, the number of outputs n and threshold k.

NOTE 2 This scheme uses an IDA to achieve optimal output size, i.e. the output size is $1/k$ of the message size[6][7].

Share space: $S^m \times N$.

5.6.3 Message sharing algorithm

Input: message $a \in G$.

Output: share vector $[a]_1,\ldots, [a]_n$.

a) Randomly select $s_1,\ldots, s_m \in X$.

b) Compute $r_i = \mathrm{DRBG}(s_i)$ for $1 \leq i \leq m$.

c) Compute $t = a - \sum_{i=1}^{m} r_i \in G$.

d) Compute the share vector $([s_i]_1,\ldots, [s_i]_n) = \mathrm{Share}(s_i)$ for $1 \leq i \leq m$.

e) Compute vector $(t'_1,\ldots, t'_n) = \mathrm{Split}(t)$.

f) Set $[a]_i = ([s_1]_i,\ldots, [s_m]_i, t'_i)$ for $1 \leq i \leq n$.

g) Output $[a]_1,\ldots, [a]_n$.

5.6.4 Message reconstruction algorithm

Input: share vector $\left([a]_{i_1},\ldots,[a]_{i_k}\right)$.

Output: message $a \in G$.

a) Compute $s_j = \mathrm{Reconst}\left(\left[s_j\right]_{i_1}, \ldots, \left[s_j\right]_{i_k}\right)$ for $1 \le j \le m$.

b) Compute $r_j = \mathrm{DRBG}(s_j)$ for $1 \le j \le m$.

c) Compute $t = \mathrm{Rec}\left(t'_{i_1}, \ldots, t'_{i_k}\right)$.

d) Compute $a = t + \sum_{i=1}^{m} r_i \in G$.

e) Output $a \in G$.

5.6.5 Properties

Confidentiality: The computational additive secret sharing scheme is computationally confidential when the receiver has less than *k* shares available.

Information rate: The information rate for the computational additive secret sharing scheme is almost *k*, which is optimal. More specifically, the size of a message is the size of one element of *G* and the size of a share is $m|S| + |N|$. If both the secret sharing scheme and the IDA achieve an optimal output size and $|S|$ is much smaller than $|G|$ then the size of a share is almost $1/k$ of the size of an element of *G*.

Homomorphic operations: The computational additive secret sharing scheme has no homomorphic operations.

Complexity: The message sharing algorithm requires *m* additions, *m* DRBG operations, *m* share operations and 1 split operation. The message reconstruction algorithm requires *m* additions, *m* DRBG operations, *m* Reconst operations and 1 Rec operation.

5.6.6 Conversion protocol

5.6.6.1 General

Although the computational additive secret sharing scheme is not homomorphic, shares of the secret sharing scheme can be converted to shares of a homomorphic secret sharing scheme. The conversion protocol[6] is described below.

5.6.6.2 Parameters

Homomorphic secret sharing scheme: a homomorphic secret sharing scheme consists of $\mathrm{HomShare}: G \to S'^{n}$, $\mathrm{HomReconst}: S'^{k} \to G$, where the share space is S', the number of shares is *n*, the threshold is *k* and homomorphic operation.

Number of seeds of the computationally additive secret sharing scheme: $m \ge k$.

Share of computationally additive secret sharing scheme: *i*-th share $\left[a\right]_i^{(\mathrm{Comp})}$ of the computational additive secret sharing scheme.

Share of homomorphic secret sharing scheme: *i*-th share $\left[a\right]_i^{(\mathrm{Hom})}$ of the homomorphic secret sharing scheme.

Homomorphic operation of the homomorphic secret sharing scheme: the homomorphic secret sharing scheme is $(+,+)$-homomorphic where addition on share vectors is performed by computing $\left[a+a'\right]_i^{(\mathrm{Hom})} = \left[a\right]_i^{(\mathrm{Hom})} + \left[a'\right]_i^{(\mathrm{Hom})}$.

NOTE The definition of (+, +)-homomorphic is shown in ISO/IEC 19592-1:2016, 5.2.2.

Parties that participate in the conversion protocol: i-th party P_i that has i-th share $\left[a\right]_i^{(\mathrm{Comp})}$.

5.6.6.3 Conversion protocol

Input: shares of computationally additive secret sharing scheme $\left[a\right]_{i_u}^{(\mathrm{Comp})}$ of P_{i_u} for $1 \le u \le m$.

Output: shares of homomorphic secret sharing scheme $\left[a\right]_i^{(\mathrm{Hom})}$ of P_i for $1 \le i \le n$.

a) Each P_{i_u} for $1 \le u \le m$ parses $\left[a\right]_{i_u}^{(\mathrm{Comp})}$ to $\left(\left[s_1\right]_{i_u},\ldots,\left[s_m\right]_{i_u}, t'_{i_u}\right)$.

b) Each P_{i_u} for $1 \le u \le m$ sends $\left[s_j\right]_{i_u}$ to P_{i_j} for $1 \le j \le m$.

c) Each P_{i_u} for $1 \le u \le m$ computes $s_u = \mathrm{Reconst}\left(\left[s_u\right]_{i_1},\ldots,\left[s_u\right]_{i_k}\right)$ and $r_u = \mathrm{DRBG}(s_u)$.

d) Each P_{i_u} for $1 \le u \le k$ sends t'_{i_u} to P_{i_1}.

e) P_{i_1} computes $t = \mathrm{Rec}\left(t'_{i_1},\ldots,t'_{i_k}\right)$.

f) P_{i_1} computes $\left(\left[t\right]_1^{(\mathrm{Hom})},\ldots,\left[t\right]_n^{(\mathrm{Hom})}\right) = \mathrm{HomShare}(t)$ and sends $\left[t\right]_i^{(\mathrm{Hom})}$ to P_i for $1 \le i \le n$.

g) Each P_{i_u} for $1 \le u \le m$ computes $\left(\left[r_u\right]_1^{(\mathrm{Hom})},\ldots,\left[r_u\right]_n^{(\mathrm{Hom})}\right) = \mathrm{HomShare}(r_u)$ and sends $\left[r_u\right]_j^{(\mathrm{Hom})}$ to P_j for $1 \le j \le n$.

h) Each P_i for $1 \le i \le n$ outputs $[a]_i^{(\mathrm{Hom})} = [t]_i^{(\mathrm{Hom})} + \sum_{j=1}^{m} [r_j]_i^{(\mathrm{Hom})}$.

5.6.6.4 Properties after running the conversion protocol

Confidentiality: The homomorphic conversion of the computational secret sharing scheme is computationally confidential when the receiver has access to less than k shares and the transactions of the conversion protocol.

Homomorphic operations: The homomorphic conversion of the computational secret sharing scheme is (+, +)-homomorphic where addition on share vectors is performed by computing $\left[a+a'\right]_i^{(\mathrm{Hom})} = \left[a\right]_i^{(\mathrm{Hom})} + \left[a'\right]_i^{(\mathrm{Hom})}$.

Annex A
(informative)

Object identifiers

This annex lists the object identifiers assigned to the secret sharing fundamental mechanisms specified in this document.

```
secret-sharing-fundamental-mechanisms {
iso(1) standard(0) secret-sharing(19592) fundamental-mechanisms(2)
asn1-module(0) object-identifiers(0) }
DEFINITIONS EXPLICIT TAGS ::= BEGIN
-- EXPORTS All; --
-- IMPORTS None; --
OID ::= OBJECT IDENTIFIER -- Alias
-- Synonyms --
id-ss-fm OID ::= {
iso(1) standard(0) secret-sharing(19592) fundamental-mechanisms(2) }
-- Assignments --
id-ss-fm-ss-1 OID ::= { id-ss-fm shamir-ss(1) }
id-ss-fm-ss-2 OID ::= { id-ss-fm ramp-ss(2) }
id-ss-fm-ss-3 OID ::= { id-ss-fm additive-general-ss(3) }
id-ss-fm-ss-4 OID ::= { id-ss-fm additive-threshold-ss(4) }
id-ss-fm-ss-5 OID ::= { id-ss-fm computational-additive-ss(5) }
-- Secret Sharing Mechanism 1 --
id-ss-fm-ss-1-1 OID ::= { id-ss-fm-ss-1 share(1) }
id-ss-fm-ss-1-2 OID ::= { id-ss-fm-ss-1 reconst(2) }
-- Secret Sharing Mechanism 2 --
id-ss-fm-ss-2-1 OID ::= { id-ss-fm-ss-2 share(1) }
id-ss-fm-ss-2-2 OID ::= { id-ss-fm-ss-2 reconst(2) }
-- Secret Sharing Mechanism 3 --
id-ss-fm-ss-3-1 OID ::= { id-ss-fm-ss-3 share(1) }
id-ss-fm-ss-3-2 OID ::= { id-ss-fm-ss-3 reconst(2) }
-- Secret Sharing Mechanism 4 --
```

```
id-ss-fm-ss-4-1 OID ::= { id-ss-fm-ss-4 share(1) }

id-ss-fm-ss-4-2 OID ::= { id-ss-fm-ss-4 reconst(2) }

-- Secret Sharing Mechanism 5 --

id-ss-fm-ss-5-1 OID ::= { id-ss-fm-ss-5 share(1) }

id-ss-fm-ss-5-2 OID ::= { id-ss-fm-ss-5 reconst(2) }

id-ss-fm-ss-5-3 OID ::= { id-ss-fm-ss-5 convert(3) }

END -- secret-sharing-fundamental-mechanisms --
```

Annex B
(informative)

Numerical examples

B.1 Shamir secret sharing scheme

Parameters:

Finite field K is a finite field of prime order $p = 2^{61} - 1$.

$(k,n) = (2,3)$.

$(x_1,x_2,x_3) = (2,3,4)$.

Message: a = “abcdef” = `0x 00006162 63646566`

Shares:

$[a]_1$ = `0x 099634bb be0a753d`

$[a]_2$ = `0x 1e611e68 6b5d7d28`

$[a]_3$ = `0x 132c0815 18b08514`

Random coefficient: r = `0x 14cae9ac ad5307eb`

B.2 Ramp Shamir secret sharing scheme

Parameters:

Finite field K is a finite field of prime order $p = 2^{61} - 1$.

$(k,L,n) = (3,2,5)$.

$(x_1,x_2,x_3,x_4,x_5) = (2,3,4,5,6)$.

Message: a = “abcdef” = `0x 00006162 63646566`

a_1 = “abc” = `0x 00616263`

a_2 = “def” = `0x 00646566`

Shares:

$[a]_1$ = `0x 02d2614f 437c38a3`

$[a]_2$ = `0x 06595af2 56c72c5a`

$[a]_3$ = `0x 0b49853d 0b3b25cb`

$[a]_4$ = `0x 11a2e02f 60d824f6`

$[a]_5$ = `0x 19656bc9 579e29db`

Random coefficient: r_2 = `0x 00b49853 d09482dd`

B.3 Additive secret sharing scheme for a general adversary structure

Parameters:

Finite cyclic group G is the additive group of a finite field of prime order $p = 2^{61} - 1$.

$\boldsymbol{A} = \{\{134\}, \{023\}, \{24\}\}$

Message: a = “abcdef” = `0x 00006162 63646566`

Shares:

$[a]_0 = \{ r_{\{134\}} = \texttt{0x 044d9c51 20caed38}, r_{\{24\}} = \texttt{0x 0098c62d 99061f19} \}$

$[a]_1 = \{ r_{\{023\}} = \texttt{0x 1b19fee3 a9935914}, r_{\{24\}} = \texttt{0x 0098c62d 99061f19} \}$

$[a]_2 = \{ r_{\{134\}} = \texttt{0x 044d9c51 20caed38} \}$

$[a]_3 = \{ r_{\{24\}} = \texttt{0x 0098c62d 99061f19} \}$

$[a]_4 = \{ r_{\{023\}} = \texttt{0x 1b19fee3 a9935914} \}$

B.4 Additive secret sharing scheme

Parameters:

Finite cyclic group G is the additive group of a finite field of prime order $p = 2^{61} - 1$.

$(k,n) = (2,3)$.

$\boldsymbol{A} = \{Z \mid Z \subset \{1,\ldots,n\}, |Z| = k-1\} = \{\{1\},\{2\},\{3\}\}$

Message: a = “abcdef” = `0x 00006162 63646566`

Shares:

$[a]_1 = \{ r_{\{2\}} = \texttt{0x 1a0779c3 11ad29a1}, r_{\{3\}} = \texttt{0x 16891be2 631205c6} \}$

$[a]_2 = \{ r_{\{3\}} = \texttt{0x 16891be2 631205c6}, r_{\{1\}} = \texttt{0x 0f6fcbbc eea535fd} \}$

$[a]_3 = \{ r_{\{1\}} = \texttt{0x 0f6fcbbc eea535fd}, r_{\{2\}} = \texttt{0x 1a0779c3 11ad29a1} \}$

B.5 Computational additive secret sharing scheme

Parameters:

K is the finite field of order $p = 2^{64}$ extended using an irreducible polynomial $x^{64} + x^4 + x^3 + x + 1$.

$G = K^{128}$ (1k Byte), $X = K^4$ (32 Byte).

$(k,n) = (2,3)$, $m = 2$.

DRBG is defined in NIST 800-90A (http://csrc.nist.gov/publications/nistpubs/800-90A/SP800-90A.pdf) with AES encryption and no derivation function and NIST_CTR_DRBG denotes the DRBG.

The Shamir secret sharing scheme with $(x_1, x_2, x_3) = (1, x, x+1)$ is used for sharing the seed.

The Rabin scheme (Ramp scheme with $L=k$) is the IDA scheme used for sharing the masked message.

Message in G: a = `0x`

```
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
abcdef0123456789   abcdef0123456789   abcdef0123456789   abcdef0123456789
```

Seed in X: s_1 = `0x`

```
cdc4b5134f2af920   8c7ddf2803851b08   0e5cb63689a1d274   735b58ad6cb19bf9
```

Seed in X: s_2 = `0x`

```
250b7c8e449082e8   b2373e9e02282ca8   915211fee1a3c6f8   a904c0d6243ee742
```

Output of DRBG in G: NIST_CTR_DRBG(s_1) = `0x`

```
030a80f8c062a4c5   d3f4fb3f8645089b   1c89562ee4cef02c   b00079552629ede4
250224e5dc4de95e   f98bb2daab97500e   e20bb4d65f3941ec   b293480d9d350f7d
af30c399abde695d   dd52b0a600d88bfe   0aa4b6756e5bcaca   f00d2cd9661475a7
48d5a75a22a94647   2065f73b588aa371   9f5d41516323ce90   9750a96070bd7e5c
46a3dd00cab74127   b1e22f14163a7bf9   fbe067f968e04ba5   eb29bfdecb741617
6d56ff045efa1156   2f68c8d2465a3713   7000b473acbc1f61   e4db420e15a912a8
2064f89c7cf14230   21f587a83076f308   0a79bde57e7dbcbe   8fb926af388ec37b
9fc1a8039ca9059f   20dfc7a39da417a4   ddffd1e271acc3cc   5844b6c6638b5f6b
987554732ba5914e   e096a9a2c7bafcb3   4e9198451ab18c68   ee2bf4f6cd52d4de
01c2723c15a68eb4   02b78266293d2acd   cbc3a334b664bb31   b9502be957f629e8
48c0d8e24d300520   f3da2bb77d6d6aa7   27ce5057ee960704   16855d8544ed0920
5ab6542930f298d5   20773ba8108b0bb9   6d9b1b28f924a512   679a93cd99ac0b53
8fb6631a7a746141   24528b1e6c83d4bf   8092c624ad060011   a4d4f78a44d63688
9961746e4e6df725   ca9841089c434425   77b7e5bb16ee0247   801e64f5aaa8d223
cdb2d2e4115bbe63   809b7192365cb959   631214424e4b17b4   e0f29c62bd434270
26cd29e42dd8769f   a316cec16b203b42   e114f3bfcfdb4a54   e253afb48175ab54
620e8761e062ec08   c958d82a797a8376   a5dff683356e6d58   a54938f47ed468db
851869ecec264efa   ee1947b9bfdc8fa3   a30b655021d85129   bea621c63954a6a2
e3c39f88918bd91f   795480da4977a786   565732f7740cc599   02074bf85e23c90c
93f4a5e8a988c577   ecf96b0b4f44dff8   6743ab73ae8a6244   99589a3eee412f28
3de3f591acbf711a   2430a162e26da407   120e20461bc02b18   ee68628098f3d0c6
96f56026c29a1933   582077a07f366a57   c0fa4df7e2e155e4   f764637d648dd69d
7bce96b154716ff2   14e50c325a75b20e   b3e3ccea27fc2bb4   62398716fe216db2
914400c983b0ded6   25b64dcbc5a59b0e   892c23ccd1f6abc9   e4aa0695e5d0f19a
db08a251b26202c7   1c3a87aa8ac0aedf   5c89989ad328f895   fbcf197bd67ac7f7
055eb9eb5d98c2bb   1cc157feb68c7392   be699d9937c895a1   8a3735dd371ef4dd
bf2a8c61963b25e1   01b243b3822b0c35   aab5594ef84b9f11   27b2fffda1162f55
927fd5c81f8d4b3a   a59236778c96f010   9db512b38d87b9e1   bdfc58029176a646
de553174742bbdef   472566745acc388e   7217945d2b6c62a6   b831eaae4e1962b9
fe18d9338bf23fcd   6c3f3f0c636a4102   2d00f5e4f3951f80   569635be19e92480
874d96023de4d4cb   35b62ae41f00ead1   86d4f2c5240c25d8   36832ee99ea5d8c6
70930e6247674a64   f0223b4f395d606f   4285dd47666bd8c3   8cee6f79618fff81
```

Output of DRBG in G: NIST_CTR_DRBG(s_2) = `0x`

```
5699461e53388573   30c47eb03b0e90e7   e404cc834830ba31   2da271e611a921ca
006d46655c52ca12   fb6d414a0ceb8ffc   d9fc1c27b2bdbade   367532f4e40d48e8
5ac52978c1d4c000   86c6cc84925c069b   b5064a39f449ec6c   d7d1f440a938ba6f
eb97de718c67fdd7   1d2b5dd44e3fa45a   68ddb249d8de9a1a   14bc32899a504fe1
8476f6373db40eab   76dfdc48d9397e1a   5afe8a5949057586   b8812e8bd6bc33b3
4311bf6263556bc9   96da0be69b6cb635   fcc0e65e20d953ab   7452585378ed269e
dd8f4128eee08013   3b39e3bd19f22e0c   956f401e4f00b30f   c37a3cd9f12a6c26
68d6d314d6565012   d60bc626cb0f8c30   51e6375003747f86   9fbd9c5def83b88c
cd3b420d4186b2c5   68cce1e8a9e6ceab   a0e3ab8f3bace6da   d692e9cdde6b9867
b52acfb952704a3b   978112c2f8e1ccfe   0cf05a016e258645   f4a9296ddff62f13
7fab6690c778c5b6   524140113c10e822   b0076ec3ca1018df   6438398336c673ff
0125e968fffafeb4   6b7bc5c029313343   4f38b40bdcbd541a   0060333d73d785ff
0d47e353ccdd721f   ada697375b18d141   418842edb794dc0e   11a2ae94b9550ea1
85fa37eb52ed53bd   003e2bf6261adea8   1a6cf2c99409665b   b775815 9e6d79661
b5df286af12459a4   51af54dc6323010b   5fff21f77e7ba9d5   6c5ee06b48211634
343708ccfb3bd7c4   190b269af67ec105   fee0dc005e51dd17   1d73b3786a01a4d7
c66d5da4b66585ba   893416ee4bca5bf0   0abc73225ca42893   b75c6e2576c1431b
fdddba304a1ca48c   e5b2a8bf9584d01f   359c035e14657e21   75f076cdf7fe18cc
ddaac20a16d9f735   f187062d42a91a20   297446c1f82c5eb6   010830c9fc38fd8a
856509b4204f7899   195eefa7b8441466   6c94a605ebede1dd   ec02afa14f4666eb
8a975e52e5974f5f   e6f0c0c73ee4f295   9fab5332e4ca7973   c5ba0a9d3e8f1aa0
e1bca72e8f24801c   502c76b6f597c224   718c6684e797fd36   eba690a49605d950
b5da2b6007486117   4863ed30c65a716a   8381fc0496e9de92   ebdac934d8afcdc7
a93ad96ae4b78d65   e472f11dec4a554d   033c0bc2cfecb5a0   15e9f4809d843c30
97469e7c018a9628   6d84d99413df93e4   41f48da750bb39d6   71399cb01102fd9a
c2c7e718a12e4af1   0ce81afcaa36fe07   d60910ee82bfc216   8e75ad655452ebdb
7b31d256405fdedb   c669c7c272b82b92   2b22f03f00db9c5b   3c52cfcff3708032
a7a7bd66e462c2ab   ad0efae4a4415d78   c7d67bf8cd1d5e60   ded73f4ca26da5f6
71903c60fb06112c   8c6e0ed4f66573cf   a92be21cc4b56a9c   32fcd6b416252110
ebad4cb577d398b2   0d4f95a5fd3ba752   e10bfa8342e0cc49   4686a5d5528c1231
74286d70660a77b4   dcedb52cfb1c557f   b00e3a5b2d34854d   2c313e013def2d44
3f9c551adbc1f3db   c7a03b2a7a768ca3   f1b84945f1e07a5b   7787f5e0b67654ca
```