



BSI Standards Publication

Information technology — Security techniques — Anonymous digital signatures

Part 2: Mechanisms using a group public key

National foreword

This British Standard is the UK implementation of ISO/IEC 20008-2:2013+A2:2023. It supersedes BS ISO/IEC 20008-2:2013, which is withdrawn.

The start and finish of text introduced or altered by amendment is indicated in the text by tags. Tags indicating changes to ISO/IEC text carry the number of the ISO/IEC amendment. For example, text altered by ISO/IEC amendment 1 is indicated by A1 A1.

The UK participation in its preparation was entrusted to Technical Committee IST/33/2, Cryptography and Security Mechanisms.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2023
Published by BSI Standards Limited 2023

ISBN 978 0 539 15676 8

ICS 35.030

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 November 2013.

Amendments/corrigenda issued since publication

Date	Text affected
30 June 2018	Implementation of ISO/IEC corrected text November 2017: In 6.5.4, step g) corrected and step h) added
28 February 2021	Implementation of ISO/IEC amendment 1:2021
30 June 2023	Implementation of ISO/IEC amendment 2:2023

First edition
2013-11-15

Corrected version
2017-11

**Information technology — Security
techniques — Anonymous digital
signatures —**

**Part 2:
Mechanisms using a group public key**

*Technologies de l'information — Techniques de sécurité — Signatures
numériques anonymes —*

Partie 2: Mécanismes utilisant une clé publique de groupe



Reference number
ISO/IEC 20008-2:2013(E)
Documents
AVA Document Sharing Hub
© ISO/IEC 2013



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2013, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

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	v
Introduction	vi
1 Scope	1
2 Normative references	1
3 Terms and definitions	1
4 Symbols (and abbreviated terms)	2
5 General model and requirements	3
6 Mechanisms with linking capability	4
6.1 General	4
6.2 Mechanism 1	5
6.2.1 Symbols	5
6.2.2 Key generation process	5
6.2.3 Signature process	8
6.2.4 Verification process	9
6.2.5 Linking process	10
6.2.6 Revocation process	10
6.3 Mechanism 2	10
6.3.1 Symbols	10
6.3.2 Key generation process	11
6.3.3 Signature process	13
6.3.4 Verification process	14
6.3.5 Linking process	15
6.3.6 Revocation process	15
6.4 Mechanism 3	16
6.4.1 Symbols	16
6.4.2 Key generation process	16
6.4.3 Signature process	17
6.4.4 Verification process	18
6.4.5 Linking process	19
6.4.6 Revocation process	19
6.5 Mechanism 4	20
6.5.1 Symbols	20
6.5.2 Key generation process	20
6.5.3 Signature process	22
6.5.4 Verification process	22
6.5.5 Linking process	23
6.5.6 Revocation process	23
7 Mechanisms with opening capability	23
7.1 General	23
7.2 Mechanism 5	23
7.2.1 Symbols	23
7.2.2 Key generation process	24
7.2.3 Signature process	25
7.2.4 Verification process	26
7.2.5 Opening process	26
7.2.6 Revocation process	26
7.3 Mechanism 6	27
7.3.1 Symbols	27
7.3.2 Key generation process	27
7.3.3 Signature process	28
7.3.4 Verification process	29
7.3.5 Opening process	29

7.3.6	Revocation process.....	29
8	Mechanisms with both opening and linking capabilities	29
8.1	General.....	29
8.2	Mechanism 7	30
8.2.1	Symbols.....	30
8.2.2	Key generation process.....	30
8.2.3	Signature process.....	32
8.2.4	Verification process.....	32
8.2.5	Opening process.....	33
8.2.6	Evidence evaluation process.....	33
8.2.7	Linking process.....	33
8.2.8	Revocation process.....	34
Annex A	(normative) Object identifiers.....	35
Annex B	(normative) Special hash-functions.....	37
Annex C	(informative) Security guidelines for the anonymous signature mechanisms.....	39
Annex D	(informative) Comparison of revocation mechanisms.....	42
Annex E	(informative) Numerical examples.....	45
Annex F	(informative) Proof of correct generation in Mechanism 5.....	80
Bibliography		84

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

ISO/IEC 20008-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

ISO/IEC 20008 consists of the following parts, under the general title *Information technology — Security techniques — Anonymous digital signatures*:

- *Part 1: General*
- *Part 2: Mechanisms using a group public key*

Further parts may follow.

This corrected version of ISO/IEC 20008-2:2013 incorporates the following correction:

- in [6.5.4](#), step g) has been corrected and step h) has been added consequently.

Introduction

Anonymous digital signature mechanisms are a special type of digital signature mechanism in which, given a digital signature, an unauthorized entity cannot discover the signer's identifier yet can verify that a legitimate signer has generated a valid signature.

ISO/IEC 20008 specifies anonymous digital signature mechanisms. ISO/IEC 20008-1 specifies principles and requirements for two categories of anonymous digital signatures mechanisms: signature mechanisms using a group public key, and signature mechanisms using multiple public keys. This part of ISO/IEC 20008 specifies a number of anonymous signature mechanisms of the first category.

Anonymous signature mechanisms of the first category can have capabilities for providing more information about the signer. Some have a linking capability, where two signatures signed by the same signer are linkable. Some have an opening capability, where the signature can be opened by a special entity to reveal the identity of the signer. Some have both linking and opening capabilities.

For each mechanism, the processes of opening, linking, and/or revocation are specified.

The mechanisms specified in this part of ISO/IEC 20008 use a collision-resistant hash-function. A hash-function specified in ISO/IEC 10118 is to be used.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity, and scope of these patent rights.

The holders of these patent right have assured the ISO and IEC that they are willing to negotiate licences either free of charge or under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC. Information may be obtained from:

- Electronics and Telecommunications Research Institute (ETRI)
161, Gajeong-dong, Yuseong-gu, Daejeon, Korea
- NEC Corporation
7-1, Shiba 5-chome, Minato-Ku, Toyko 108-8001, Japan

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO and/or IEC shall not be held responsible for identifying any or all such patent rights.

ISO (www.iso.org/patents) and IEC (<http://patents.iec.ch>) maintain on-line databases of patents relevant to their standards. Users are encouraged to consult the databases for the most up to date information concerning patents.

Information technology — Security techniques — Anonymous digital signatures —

Part 2: Mechanisms using a group public key

1 Scope

This part of ISO/IEC 20008 specifies anonymous digital signature mechanisms, in which a verifier makes use of a group public key to verify a digital signature.

It provides

- a general description of an anonymous digital signature mechanism using a group public key, and
- a variety of mechanisms that provide such anonymous digital signatures.

For each mechanism, this part of ISO/IEC 20008 specifies

- the process for generating group member signature keys and a group public key,
- the process for producing signatures,
- the process for verifying signatures,
- the process for opening signatures (if the mechanism supports opening),
- the process for linking signatures (if the mechanism supports linking), and
- the process for revoking group members.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 10118 (all parts), *Information technology — Security techniques — Hash-functions*

ISO/IEC 15946-5, *Information technology — Security techniques — Cryptographic techniques based on elliptic curves — Part 5: Elliptic curve generation*

ISO/IEC 18031, *Information technology — Security techniques — Random bit generation*

ISO/IEC 18032, *Information technology — Security techniques — Prime number generation*

ISO/IEC 20008-1, *Information technology — Security techniques — Anonymous digital signatures — Part 1: General*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 20008-1 and the following apply.

- 3.1**
assistant signer
entity that can help a principal signer to create anonymous signatures, but that cannot generate anonymous signatures unaided
- 3.2**
member-list
list that includes the identities of group members together with their corresponding group membership credentials
- 3.3**
principal signer
entity which is in possession of a group member signature key and which can create anonymous signatures using this key
- 3.4**
secret seed value
secret data known to a group member and used for deriving group member private keys
- 3.5**
security parameters
variables that determine the security strength of a mechanism

4 Symbols (and abbreviated terms)

For the purposes of this part of ISO/IEC 20008, the following symbols and abbreviations apply.

bsn	Linking base, either a special symbol $\perp$ or an arbitrary string.
e	A bilinear map function $e: G_1 \times G_2 \rightarrow G_T$ such that for all $P \in G_1$, $Q \in G_2$, and all positive integers a, b , the equation $e([a]P, [b]Q) = e(P, Q)^{ab}$ holds. This function is also called a pairing function.
$\mathbb{A}_2$ $F(p)$	the finite field containing exactly p elements. $\mathbb{A}_2$
$\gcd(a, b)$	The greatest common divisor of the integers a and b .
G_1	An additive cyclic group of order p over an elliptic curve.
G_2	An additive cyclic group of order p over an elliptic curve.
G_T	A multiplicative cyclic group of order p .
H	A cryptographic hash-function.
m	Message to be signed.
n	An RSA modulus where $n = pq$.
O_E	The elliptic curve point at infinity.
p	A prime number.
P_1	Generator of G_1 .
P_2	Generator of G_2 .
q	A prime number.
Q_1+Q_2	The elliptic curve sum of points Q_1 and Q_2 .
$QR(n)$	The group of quadratic residues modulo n .

Z_n^*	The multiplicative group of invertible elements in Z_n .
Z_p	The set of integers in $[0, p-1]$.
Z_p^*	The set of integers in $[1, p-1]$.
$(a p)$	The Legendre symbol of a and p where a is an integer and p is an odd prime number.
$[n]P$	Multiplication operation that takes a positive integer n and a point P on the elliptic curve E as input and produces as output another point Q on the curve E , where $Q = [n]P = P + P + \dots + P$, i.e. the sum of n copies of P . The operation satisfies $[0]P = O_E$ and $[-n]P = [n](-P)$.
$[x, y]$	The set of integers from x to y inclusive, if x, y are integers satisfying $x \leq y$.
$ $	$X Y$ is used to mean the result of the concatenation of data items X and Y in the order specified. In cases where the result of concatenating two or more data items is signed as part of one of the mechanisms specified in this part of ISO/IEC 20008, this result shall be composed so that it can be uniquely resolved into its constituent data strings, i.e. so that there is no possibility of ambiguity in interpretation. This latter property could be achieved in a variety of different ways, depending on the application. For example, it could be guaranteed by (a) fixing the length of each of the substrings throughout the domain of use of the mechanism, or (b) encoding the sequence of concatenated strings using a method that guarantees unique decoding, e.g. using the distinguished encoding rules defined in ISO/IEC 8825-1. ^[1]

5 General model and requirements

This clause specifies the general model and requirements for the anonymous digital signature mechanisms specified in this part of ISO/IEC 20008. Some of the contents of this clause are taken from Part 1 of this international standard. In addition, specific requirements applying to mechanisms using a group public key are addressed.

An anonymous digital signature mechanism using a group public key involves a group and a set of group members. Each group shall possess a group membership issuer. There may also be a group membership opener and/or a group signature linker, depending on the mechanism. Multiple entities may function in the role of a group membership opener or a group signature linker. The level of anonymity of the mechanism depends on the anonymity strength (i.e., the size of the group), whether there is an opening capability, whether there is a linking capability, how revocation is done, whether the issuer knows the private keys, and the likelihood of compromise of a private key.

Such an anonymous digital signature mechanism is defined by the specification of the following processes:

- key generation process,
- signature process,
- verification process,
- opening process (if the mechanism supports opening),
- linking process (if the mechanism supports linking), and
- revocation process.

The anonymous digital signature mechanisms using a group public key specified in this part of ISO/IEC 20008 involve a range of types of entity. Some of these entities exist in every mechanism whereas others exist only in some mechanisms. These entities are as follows:

- **Signer:** a signer is an entity that generates a digital signature. In some mechanisms, a signer role is split between two entities. For example, in direct anonymous attestation mechanisms, the signer role is split between a principal signer with limited computational and storage capability, e.g. a trusted platform module (TPM), and an assistant signer with more computational power but less security tolerance, e.g. an ordinary computer platform (namely the Host with an embedded TPM).

- **Verifier:** a verifier is an entity that verifies a digital signature.
- **Group membership issuer:** a group membership issuer is an entity that issues a group membership credential to a signer. This entity exists in all the mechanisms.
- **Group membership opener:** a group membership opener is an entity who can identify the signer from a signature. This entity exists in some of the mechanisms.
- **Group signature linker:** a group signature linker is an entity that checks whether two signatures have been generated by the same signer with a linking key or a linking base. This entity exists in some of the mechanisms.

In order to use any of the mechanisms specified in this part of ISO/IEC 20008, the following requirements shall be met:

- Each entity involved in an anonymous digital signature mechanism is aware of a common set of group public parameters, which are used to compute a variety of functions in the mechanism.
- Each verifier has access to an authentic copy of the group public key.
- An authentic channel is required between a signer and a group membership issuer during the process of issuing group member signature key. This ensures that the group membership issuer is able to provide the group member signature key only to a legitimate group member.
- A collision-resistant hash function such as one of those specified in ISO/IEC 10118 shall be used.
- A robust random bit generator such as one of those specified in ISO/IEC 18031 shall be used.
- A robust prime number generator such as one of those specified in ISO/IEC 18032 shall be used.
- A robust elliptic curve generator such as one of those specified in ISO/IEC 15946-5 shall be used in some mechanisms.

A1 Annex A lists the object identifiers which shall be used to identify the mechanisms specified in this document.

Annex B defines the special hash-functions which shall be used for the mechanisms specified in this document.

Annex C defines the security guidelines for the anonymous signature mechanisms specified in this document.

Annex D provides a comparison of the revocation mechanisms specified in this document.

Annex E provides numerical examples for each digital signature mechanism specified in this document.

Annex F provides a technique for proof of correct generation for Mechanism 5 of this document. **A1**

6 Mechanisms with linking capability

6.1 General

A2 This clause specifies five digital signature mechanisms with linking capability.

NOTE 1 In the literature, the mechanism of 6.2 is called a list signature scheme, the mechanism of 6.6 is called a pre-DAA scheme and the mechanisms of 6.3, 6.4 and 6.5 are called DAA schemes. The mechanisms given in 6.2, 6.4, 6.5 and 6.6 are based on schemes originally specified in References [9], [6], [11] and [22] respectively, in which security proofs can also be found. The mechanism in 6.3 is based on a scheme in Reference [3] which is a minor modification of the scheme in Reference [4]; the associated security analysis is given in the full version of Reference [4]. **A2**

NOTE 2 For certain applications such as attestation, a message to be signed may be hashed and/or concatenated with additional information before being input to the signature process of one of the anonymous digital signature mechanisms specified in this clause.

6.2 Mechanism 1

6.2.1 Symbols

The following symbols apply in the specification of this mechanism.

- $l_p, k, l_x, l_e, l_E, l_X, \varepsilon$: security parameters.
- p', q', e : prime numbers.
- $a, a_0, g, h, b, C_1, D, C_2, d', d_1, d_2, t', t_1, t_2, A, f, T_1, T_2, T_3, T_4, d_3, d_4, d_5, t_3, t_4, t_5$: integers in $QR(n)$.
- x', α, β : integers in $[0, 2^{l_x} - 1]$.
- w_1, w_2, w_3 : integers in $[0, 2^{2l_p} - 1]$.
- $\hat{c}, \dot{c}, c', c, c'', c'''$: k -bit integers.
- $\check{r}$: $(2l_p + 1)$ -bit integer.
- $t_1, \hat{s}_1, r', r_1, r_2$: $(\varepsilon \cdot (l_x + k))$ -bit integers.
- $t_2, \hat{s}_2$: $(\varepsilon \cdot (2l_p + k + 1))$ -bit integers.
- x : $(l_x + 1)$ -bit integer.
- r_3 : $(\varepsilon \cdot (l_x + 2l_p + k + 1))$ -bit integer.
- s_0, s_1, s_2, s' : integers in $[-2^{l_x+k}, 2^{\varepsilon(l_x+k)} - 1]$.
- s_3 : integer in $[-2^{l_x+2l_p+k+1}, 2^{\varepsilon(l_x+2l_p+k+1)} - 1]$.
- r_4, r_5 : $(\varepsilon \cdot (2l_p + k))$ -bit integers.
- r_9, r_{10} : $(\varepsilon \cdot (2l_p + l_e + k))$ -bit integers.
- s_4, s_5 : integers in $[-2^{2l_p+k}, 2^{\varepsilon(2l_p+k)} - 1]$.
- s_9, s_{10} : integers in $[-2^{2l_p+l_e+k}, 2^{\varepsilon(2l_p+l_e+k)} - 1]$.
- H : a hash function that outputs k -bit message digest.
- H_I : a hash function that outputs $(2l_p)$ -bit message digest.

6.2.2 Key generation process

The key generation process has two parts: a setup process and a group membership issuing process. The setup process is executed by the group membership issuer to create the group public parameter, group public key, and group membership issuing key. The group membership issuing process is an interactive protocol running between the group membership issuer and a group member to create a unique group member signature key for the group member.

The setup process takes the following steps by the group membership issuer:

- a) Choose the following parameters: $l_p, k, l_x, l_e, l_E, l_X, \varepsilon$.
- b) Choose an RSA modulus $n = pq$ with $p = 2p' + 1, q = 2q' + 1$ such that p, q, p', q' are all primes and p' as well as q' have l_p bits.
- c) Choose a random generator a of the group of quadratic residues modulo n by performing the following steps:
 - 1) Choose a random integer g in Z_n^* such that $\gcd(g+1, n) = 1$ and $\gcd(g-1, n) = 1$.

- 2) Compute $a = g^2 \pmod n$.
- d) Choose a random generator a_0 of $QR(n)$ different from a .
- e) Choose a random generator g of $QR(n)$ different from a and a_0 .
- f) Choose a random generator h of $QR(n)$ different from a , a_0 and g .
- g) Choose a random generator b of $QR(n)$ different from a , a_0 , g and h .
- h) The group membership issuer chooses two hash functions $H: \{0, 1\}^* \rightarrow \{0, 1\}^k$ and $H_r: \{0, 1\}^* \rightarrow \{0, 1\}^{2lp}$. An example of how to construct H_r is provided in [Annex B](#).
- i) Output the following:
 - group public parameter = $(l_p, k, l_x, l_e, l_E, l_X, \varepsilon)$,
 - group public key = (n, a, a_0, g, h, b) ,
 - group membership issuing key = (p', q') .

NOTE An example of recommended parameters is provided in Annex [C.2](#).

The group membership issuing process may require a secure and authentic channel between the member and the group membership issuer to prevent the group membership credential from being observed by an eavesdropper. How to establish such a channel is out of scope of this mechanism. The group membership issuing process is as follows:

- a) The group member chooses a random integer $x' \in [0, 2^{l_x} - 1]$.
- b) The member chooses a random integer $\check{r} \in [0, 2n - 1]$.
- c) The member computes $C_1 = g^{x'} h^{\check{r}} \pmod n$.
- d) The member generates a proof of knowledge U of the representation $(x', \check{r})$ of C_1 in the bases g and h by performing the following steps:
 - 1) The member chooses a random integer $t_1 \in [0, 2^{\varepsilon(l_x+k)} - 1]$.
 - 2) The member chooses a random integer $t_2 \in [0, 2^{\varepsilon(2lp+k+1)} - 1]$.
 - 3) The member computes $D = g^{t_1} h^{t_2} \pmod n$.
 - 4) The member computes $\hat{c} = H(g \parallel h \parallel C_1 \parallel D)$.
 - 5) The member computes $\hat{s}_1 = t_1 - \hat{c} x'$.
 - 6) The member computes $\hat{s}_2 = t_2 - \hat{c} \check{r}$.
 - 7) $U = (\hat{c}, \hat{s}_1, \hat{s}_2)$.
- e) The member sends C_1 and U to the group membership issuer.
- f) The group membership issuer receives C_1 and U from the member.
- g) The group membership issuer verifies that C_1 belongs to $QR(n)$ by performing the following step:
 - 1) The group membership issuer checks that $(C_1|p) = 1$ and that $(C_1|q) = 1$. If either of these verifications fails, the group membership issuer outputs Reject and stops.
- h) The group membership issuer verifies the proof of knowledge U by performing the following steps:
 - 1) The group membership issuer computes $D' = g^{\hat{s}_1} h^{\hat{s}_2} C_1^{\hat{c}} \pmod n$.
 - 2) The group membership issuer computes $\hat{c} = H(g \parallel h \parallel C_1 \parallel D')$.

- 3) The group membership issuer checks that $\hat{c} = \hat{c}$, $\hat{s}_1$ belongs to $[-2^{lx+k}, 2^{\varepsilon(lx+k)}-1]$ and $\hat{s}_2$ belongs to $[-2^{2lp+k+1}, 2^{\varepsilon(2lp+k+1)}-1]$. If any of these verifications fails, the group membership issuer outputs Reject and stops.
- i) The group membership issuer chooses a random odd integer $\alpha \in [0, 2^{lx}-1]$.
- j) The group membership issuer chooses a random integer $\beta \in [0, 2^{lx}-1]$.
- k) The group membership issuer sends α and β to the member.
- l) The member receives α and β from the group membership issuer.
- m) The member computes $x = 2^{lx} + (\alpha x' + \beta \pmod{2^{lx}})$.
- n) The member computes $C_2 = a^x \pmod{n}$.
- o) The member computes $v = (\alpha x' + \beta) \mid 2^{lx}$.
- p) The member generates a proof of knowledge V of the discrete logarithm x of C_2 in base a by performing the following steps:
 - 1) The member chooses a random integer $r' \in [0, 2^{\varepsilon(lx+k)}-1]$.
 - 2) The member computes $d' = a^{r'} \pmod{n}$.
 - 3) The member computes $c' = H(a \parallel g \parallel C_2 \parallel d')$.
 - 4) The member computes $s' = r' - c'(x - 2^{lx})$.
 - 5) The member set $V = (c', s')$.
- q) The member generates a proof of knowledge W by performing the following steps:
 - 1) The member chooses a random integer $r_1 \in [0, 2^{\varepsilon(lx+k)}-1]$.
 - 2) The member chooses a random integer $r_2 \in [0, 2^{\varepsilon(lx+k)}-1]$.
 - 3) The member chooses a random integer $r_3 \in [0, 2^{\varepsilon(lx+2lp+k+1)}-1]$.
 - 4) The member computes $d_1 = a^{r_1} \pmod{n}$.
 - 5) The member computes $d_2 = g^{r_1}(g^l)^{r_2}h^{r_3} \pmod{n}$ where $l = 2^{lx}$.
 - 6) The member computes $c = H(a \parallel g \parallel h \parallel C_1 \parallel C_2 \parallel d_1 \parallel d_2)$.
 - 7) The member computes $s_1 = r_1 - c(x - 2^{lx})$.
 - 8) The member computes $s_2 = r_2 - cv$.
 - 9) The member computes $s_3 = r_3 - c\alpha\checkmark$.
 - 10) The member sets $W = (c, s_1, s_2, s_3)$.
- r) The member sends C_2 , V and W to the group membership issuer.
- s) The group membership issuer receives C_2 , V and W from the member.
- t) The group membership issuer checks that C_2 belongs to $QR(n)$ by performing the following step:
 - 1) The group membership issuer checks that $(C_2|p) = 1$ and that $(C_2|q) = 1$. If any of these verifications fails, the group membership issuer outputs Reject and stops.
- u) The group membership issuer verifies the proof of knowledge V by performing the following steps:
 - 1) The group membership issuer computes $s_0 = s' - c' 2^{lx}$.

- 2) The group membership issuer computes $t' = C_2 c' a^{s_0} \pmod n$.
- 3) The group membership issuer computes $c'' = H(a \parallel g \parallel C_2 \parallel t')$.
- 4) The group membership issuer checks that $c'' = c'$ and that $s' \in [-2^{lx+k}, 2^{\varepsilon(lx+k)}-1]$. If any of these verifications fails, the group membership issuer outputs Reject and stops.
- v) The group membership issuer verifies the proof of knowledge W by performing the following steps:
 - 1) The group membership issuer computes $t_1 = (C_2/a^L) c a^{s_1} \pmod n$ where $L = 2^{lx}$.
 - 2) The group membership issuer computes $t_2 = (C_1 \alpha g^{\beta}) c g^{s_1} (g^l)^{s_2} h^{s_3} \pmod n$ where $l = 2^{lx}$.
 - 3) The group membership issuer computes $c''' = H(a \parallel g \parallel h \parallel C_1 \parallel C_2 \parallel t_1 \parallel t_2)$.
 - 4) The group membership issuer checks that: $c''' = c$, s_1 belongs to $[-2^{lx+k}, 2^{\varepsilon(lx+k)}-1]$, s_2 belongs to $[-2^{lx+k}, 2^{\varepsilon(lx+k)}-1]$ and that s_3 belongs to $[-2^{lx+2lp+k+1}, 2^{\varepsilon(lx+2lp+k+1)}-1]$. If any of these verifications fails, the group membership issuer outputs Reject and stops.
- w) The group membership issuer chooses a random prime $e \in [2^{le} - 2^{le} + 1, 2^{le} + 2^{le} - 1]$.
- x) The group membership issuer computes $\bar{d}_1 = 1/e \pmod{p'q'}$.
- y) The group membership issuer computes $A = (a_0 C_2)^{\bar{d}_1} \pmod n$.
- z) The group membership issuer stores $(A, e, Member)$ in member-list LIST.
- aa) The group membership issuer sends A and e to the member.
- bb) The member receives A and e from the group membership issuer.
- cc) The member checks that $A^e = a_0 a^x \pmod n$.
- dd) The group member signature key of the signer is (A, e, x) , in which x is the group member private key and (A, e) is the group membership credential.

6.2.3 Signature process

On input of a group public key (n, a, a_0, g, h, b) , a group member signature key (A, e, x) , a linking base bsn , and a message $m \in \{0, 1\}^*$ to be signed, the signature process takes the following steps below. The linking base is used for the linking capability. It is chosen by group membership issuer or any other trusted authorities.

- a) The member computes $f = (H_r(bsn))^2 \pmod n$.
- b) The member chooses a random integer $w_1 \in [0, 2^{2lp} - 1]$.
- c) The member chooses a random integer $w_2 \in [0, 2^{2lp} - 1]$.
- d) The member chooses a random integer $w_3 \in [0, 2^{2lp} - 1]$.
- e) The member computes $T_1 = Ab^{w_1} \pmod n$.
- f) The member computes $T_2 = g^{w_1} h^{w_2} \pmod n$.
- g) The member computes $T_3 = g^e h^{w_3} \pmod n$.
- h) The member computes $T_4 = f^x \pmod n$.
- i) The member chooses a random integer $r_1 \in [0, 2^{\varepsilon(le+k)}-1]$.
- j) The member chooses a random integer $r_2 \in [0, 2^{\varepsilon(lx+k)}-1]$.
- k) The member chooses a random integer $r_3 \in [0, 2^{\varepsilon(2lp+k)}-1]$.

- l) The member chooses a random integer $r_4 \in [0, 2^{\varepsilon(2lp+k)}-1]$.
- m) The member chooses a random integer $r_5 \in [0, 2^{\varepsilon(2lp+k)}-1]$.
- n) The member chooses a random integer $r_9 \in [0, 2^{\varepsilon(2lp+le+k)}-1]$.
- o) The member chooses a random integer $r_{10} \in [0, 2^{\varepsilon(2lp+le+k)}-1]$.
- p) The member computes $d_1 = T_1 r_1 / (a r^2 b r^9) \pmod n$.
- q) The member computes $d_2 = T_2 r_1 / (g r^9 h r^{10}) \pmod n$.
- r) The member computes $d_3 = g r^3 h r^4 \pmod n$.
- s) The member computes $d_4 = g r^1 h r^5 \pmod n$.
- t) The member computes $d_5 = f r^2 \pmod n$.
- u) The member computes $c = H(a \parallel a_0 \parallel g \parallel h \parallel T_1 \parallel T_2 \parallel T_3 \parallel T_4 \parallel d_1 \parallel d_2 \parallel d_3 \parallel d_4 \parallel d_5 \parallel m)$.
- v) The member computes $s_1 = r_1 - c(e - 2^{lE})$.
- w) The member computes $s_2 = r_2 - c(x - 2^{lX})$.
- x) The member computes $s_3 = r_3 - c w_1$.
- y) The member computes $s_4 = r_4 - c w_2$.
- z) The member computes $s_5 = r_5 - c w_3$.
- aa) The member computes $s_9 = r_9 - c e w_1$.
- bb) The member computes $s_{10} = r_{10} - c e w_2$.
- cc) The member sets the signature as $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$.

6.2.4 Verification process

On input of a message m , a linking base bsn , a signature $(c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$, a group public key (n, a, a_0, g, h, b) , the verification process takes the following steps:

- a) The verifier computes $f = (H_r(bsn))^2 \pmod n$.
- b) The verifier computes $t_1 = a_0^c T_1^{s_1 - c l'} / (a^{s_2 - c L} b^{s_9}) \pmod n$ where $l' = 2^{lE}$ and $L = 2^{lX}$.
- c) The verifier computes $t_2 = T_2^{s_1 - c l'} / (g^{s_9} h^{s_{10}}) \pmod n$ where $l' = 2^{lE}$.
- d) The verifier computes $t_3 = T_2^c g^{s_3} h^{s_4} \pmod n$.
- e) The verifier computes $t_4 = T_3^c g^{s_1 - c l'} h^{s_5} \pmod n$ where $l' = 2^{lE}$.
- f) The verifier computes $t_5 = T_4^c f^{s_2 - c L} \pmod n$ where $L = 2^{lX}$.
- g) The verifier computes $c' = H(a \parallel a_0 \parallel g \parallel h \parallel T_1 \parallel T_2 \parallel T_3 \parallel T_4 \parallel t_1 \parallel t_2 \parallel t_3 \parallel t_4 \parallel t_5 \parallel m)$.
- h) If $c' = c$, s_1 belongs to $[-2^{le+k}, 2^{\varepsilon(le+k)}-1]$, s_2 belongs to $[-2^{lx+k}, 2^{\varepsilon(lx+k)}-1]$, s_3 belongs to $[-2^{2lp+k}, 2^{\varepsilon(2lp+k)}-1]$, s_4 belongs to $[-2^{2lp+k}, 2^{\varepsilon(2lp+k)}-1]$, s_5 belongs to $[-2^{2lp+k}, 2^{\varepsilon(2lp+k)}-1]$, s_9 belongs to $[-2^{2lp+le+k}, 2^{\varepsilon(2lp+le+k)}-1]$, s_{10} belongs to $[-2^{2lp+le+k}, 2^{\varepsilon(2lp+le+k)}-1]$, then return 1(valid).
- i) Else return 0 (invalid).

6.2.5 Linking process

Given two valid signatures $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ and $\sigma' = (c', s_1', s_2', s_3', s_4', s_5', s_9', s_{10}', T_1', T_2', T_3', T_4')$ computed using a linking base bsn , the linking process takes the following step:

- a) If $T_4 = T_4'$, output 1 (linked), otherwise, output 0 (not linked).

6.2.6 Revocation process

Details of the revocation process in this mechanism are surveyed in [10]. There are two types of revocation (private key revocation and verifier blacklist revocation) supported in this mechanism. Private key revocation can be either a global revocation or a local revocation. Verifier blacklist revocation is a local revocation.

Private key revocation:

- If a group member signature key (A, e, x) is compromised, the group membership issuer or a verifier puts x into a revocation list RL of this type.
- Given a valid signature $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ computed using a linking base bsn and a revocation list RL of this type, a verifier can check revocation of this signature as follows: For each $x' \in \text{RL}$, verify $T_4 \neq (H_\Gamma(bsn))^{2x'} \pmod{n}$. If any of the verification fails, output 0 (revoked), otherwise, output 1 (valid).

NOTE The private key revocation works only if the group membership issuer or the verifier has learned the group member signature keys of the compromised group members.

Verifier blacklist revocation:

- If signatures were computed using a linking base bsn , and a verifier can build its own revocation list RL corresponding to bsn . If the verifier wants to blacklist the signer of a signature $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$, she puts T_4 into a revocation list RL of this type.
- Given a valid signature $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ computed using a linking base bsn and a revocation list RL of this type, a verifier can check revocation of this signature as follows: For each $T_4' \in \text{RL}$, verify $T_4 \neq T_4'$. If any of the verification fails, output 0 (revoked), otherwise, output 1 (valid).

NOTE In order to use verifier blacklist revocation in this mechanism, a signer is required to use a specific linking base for each verifier. The value of the linking base could, for example, be chosen by the verifier or agreed in advance by the signer and verifier.

6.3 Mechanism 2

6.3.1 Symbols

The following symbols apply in the specification of this mechanism.

- $l_n, l_f, l_e, l'_e, l_v, l_\emptyset, l_H, l_r, l_s, l_\Gamma, l_\rho$: security parameters.
- p', q', ρ, Γ, e : prime numbers.
- $g', g, h, S, Z, R_0, R_1, U, U', A, A', T, T_t'$: integers in $\text{QR}(n)$.
- $x_0, x_1, x_z, x_s, x_h, x_g, s_e, r_e$: integers in $[1, p'q']$.
- $\gamma, J_l, K_l, K_l', J, K, J', K'$: integers whose multiplicative order modulo Γ is ρ .
- f, f' : integers in $[0, \rho-1]$.
- f_0, f_1 : l_f -bit integers.

- c_h, c, c', n_l, n_v : l_H -bit integers.
- n_T, n_H : $l_\emptyset$ -bit integers.
- t, t_2, r_f : $(2l_f + l_\emptyset + l_H + 1)$ -bit integers.
- t_1 : $(l_e + l_H)$ -bit integer.
- r_0, r_1 : $(l_f + l_\emptyset + l_H)$ -bit integers.
- r_v : $(l_v + l_\emptyset + l_H)$ -bit integer.
- r_{v^*} : $(l_e + l_n + 2l_\emptyset + l_H + 1)$ -bit integer.
- $r_{v'}$: $(l_n + 2l_\emptyset + l_H)$ -bit integer.
- s_v : $(l_n + 2l_\emptyset + l_H + 1)$ -bit integer.
- s_0, s_1 : $(l_f + l_\emptyset + l_H + 1)$ -bit integers.
- v', w : $(l_n + l_\emptyset)$ -bit integers.
- v^* : $(l_v - 1)$ -bit integer.
- v'' : l_v -bit integer.
- v : $(l_n + l_\emptyset + l_v + 1)$ -bit integer.
- bsn_I : a linking base of the group membership issuer.
- H_I : a hash function that outputs $(l_I + l_\emptyset)$ -bit message digest.

6.3.2 Key generation process

The key generation process has two parts: a setup process and a group membership issuing process. The setup process is executed by the group membership issuer to create the group public parameter, group public key, and group membership issuing key. The group membership issuing process is an interactive protocol running between the group membership issuer and a group member to create a unique group member signature key for the group member. Revocation check during the group membership issuing process is required to prevent an attacker from re-enrolling a revoked group member private key.

The setup process takes the following steps by the group membership issuer:

- a) Choose the following parameters: $l_n, l_f, l_e, l'_e, l_v, l_\emptyset, l_H, l_r, l_s, l_I, l_\rho$.
- b) Choose two hash functions $H: \{0, 1\}^* \rightarrow \{0, 1\}^{l_H}$ and $H_I: \{0, 1\}^* \rightarrow \{0, 1\}^{l_I + l_\emptyset}$. An example of how to construct H_I is provided in [Annex B](#).
- c) Choose an RSA modulus $n = pq$ with $p = 2p' + 1, q = 2q' + 1$ such that p, q, p', q' are all primes and n has l_n bits.
- d) Choose a random integer g' in $\text{QR}(n)$.
- e) Choose random integers $x_0, x_1, x_z, x_s, x_h, x_g$ in $[1, p'q']$.
- f) Compute $g = (g')^{x_g} \bmod n, h = (g')^{x_h} \bmod n, S = h^{x_s} \bmod n$.
- g) Compute $Z = h^{x_z} \bmod n, R_0 = S^{x_0} \bmod n, R_1 = S^{x_1} \bmod n$.
- h) Choose a random prime ρ of l_ρ bits.
- i) Choose a random prime Γ of l_I bits such that $\Gamma - 1$ is a multiple of ρ and $(\Gamma - 1)/\rho$ is not a multiple of ρ .
- j) Choose a random γ , a number whose multiplicative order modulo Γ is ρ .

k) Output the following:

- group public parameter = $(l_n, l_f, l_e, l'_e, l_v, l_\emptyset, l_H, l_r, l_s, l_\Gamma, l_\rho, H, H_\Gamma)$,
- group public key = $(n, g', g, h, S, Z, R_0, R_1, \gamma, \Gamma, \rho)$,
- group membership issuing key = (p', q') .

NOTE An example of recommended parameters is provided in Annex C.2.

The group membership issuing process requires a secure and authentic channel between the principal signer and the group membership issuer. How to establish such a channel is out scope of this mechanism. The group membership issuing process includes the following steps:

- a) The principal signer and the group membership issuer agree on a linking base bsn_I .
- b) The assistant signer computes $J_I = (H_\Gamma(1 \parallel bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$ and sends J_I to the principal signer.

NOTE As discussed in [19], holding the signer anonymity requires that the value bsn_I is never used as bsn in any DAA signatures. In an application where this requirement is not guaranteed, changing $J_I = (H_\Gamma(1 \parallel bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$ to $J_I = (H_\Gamma(0 \parallel bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$ can avoid this potential problem.

- c) The principal signer checks that $(J_I)^\rho \equiv 1 \pmod{\Gamma}$.
- d) The principal signer chooses a random $f \in [0, \rho-1]$ or derives f from its secret seed value.
- e) The principal signer computes $f_1 = \lfloor f/2^{lf} \rfloor$.
- f) The principal signer computes $f_0 = f - 2^{lf} \cdot f_1$. The (f_0, f_1) pair is part of the group member private key.
- g) The principal signer randomly picks an integer v' of $(l_n + l_\emptyset)$ -bit.
- h) The principal signer computes $U = (R_0^{f_0} \cdot R_1^{f_1} \cdot S^{v'}) \bmod n$.
- i) The principal signer computes $K_I = (J_I)^f \bmod \Gamma$.
- j) The principal signer sends U and K_I to the assistant signer who forwards them to the group membership issuer.

NOTE After the above step, the group membership issuer can compute $f = f_0 + f_1 \cdot 2^{lf}$ for each (f_0, f_1) on the revocation list and verify $K_I \neq J_I^f \bmod \Gamma$. If the signer has been revoked, the group membership issuer aborts the group membership issuing process.

- k) The principal signer randomly picks two integers r_0, r_1 of $(l_f + l_\emptyset + l_H)$ -bit.
- l) The principal signer randomly picks an integer $r_{v'}$ of $(l_n + 2l_\emptyset + l_H)$ -bit.
- m) The principal signer computes $U' = (R_0^{r_0} \cdot R_1^{r_1} \cdot S^{r_{v'}}) \bmod n$.
- n) The principal signer computes $r_f = r_0 + r_1 \cdot 2^{lf}$.
- o) The principal signer computes $K_I' = (J_I)^{r_f} \bmod \Gamma$.
- p) The signer sends U' and K_I' to the assistant signer.
- q) The group membership issuer chooses a random $n_I \in \{0, 1\}^{l_H}$ and sends n_I to the assistant signer.
- r) The assistant signer computes $c_h = H(n \parallel R_0 \parallel R_1 \parallel S \parallel U \parallel K_I \parallel U' \parallel K_I' \parallel n_I)$.
- s) The assistant signer sends c_h to the principal signer.
- t) The principal signer chooses a random nonce $n_T \in \{0, 1\}^{l_\emptyset}$.
- u) The principal signer computes $c = H(c_h \parallel n_T)$.

- v) The principal signer computes $s_{v'} = r_{v'} + c \cdot v'$, $s_0 = r_0 + c \cdot f_0$, $s_1 = r_1 + c \cdot f_1$.
- w) The principal signer sends $(c, n_T, s_0, s_1, s_{v'})$ to the assistant signer.
- x) The assistant signer forwards $(c, n_T, s_0, s_1, s_{v'})$ to the group membership issuer.
- y) The group membership issuer verifies that s_0 and s_1 are at most $(l_f + l_\emptyset + l_H + 1)$ -bit integers.
- z) The group membership issuer verifies that $s_{v'}$ is an at most $(l_n + 2l_\emptyset + l_H + 1)$ -bit integer.
- aa) The group membership issuer computes $U' = (U^{-c} \cdot R_0^{s_0} \cdot R_1^{s_1} \cdot S^{s_{v'}}) \bmod n$.
- bb) The group membership issuer computes $t = s_0 + s_1 \cdot 2^{l_f}$.
- cc) The group membership issuer computes $K_I' = (K_I^{-c} \cdot J_I^t) \bmod \Gamma$.
- dd) The group membership issuer verifies that $c = H(H(n \parallel R_0 \parallel R_1 \parallel S \parallel U \parallel K_I \parallel U' \parallel K_I' \parallel n_I) \parallel n_T)$.
- ee) The group membership issuer randomly chooses v^* of $(l_v - 1)$ -bit.
- ff) The group membership issuer randomly choose a prime e from $[2^{le-1}, 2^{le-1} + 2^{l'e-1}]$.
- gg) The group membership issuer computes $v'' = v^* + 2^{lv-1}$.
- hh) The group membership issuer computes $A = (Z \cdot U^{-1} \cdot S^{-v''})^{1/e} \bmod n$.
- ii) The assistant signer chooses a random integer $n_H \in \{0, 1\}^{l_\emptyset}$ and sends to the group membership issuer.
- jj) The group membership issuer randomly chooses r_e from $[0, p' \cdot q']$.
- kk) The group membership issuer computes $A' = (Z \cdot U^{-1} \cdot S^{-v''})^{r_e} \bmod n$.
- ll) The group membership issuer computes $c' = H(n \parallel Z \parallel S \parallel U \parallel v'' \parallel A \parallel A' \parallel n_H)$.
- mm) The group membership issuer computes $s_e = (r_e - c'/e) \bmod p' \cdot q'$.
- nn) The group membership issuer sends c' , s_e , and (A, e, v'') to the assistant signer.
- oo) The assistant signer verifies that e is a prime in $[2^{le-1}, 2^{le-1} + 2^{l'e-1}]$.
- pp) The assistant signer computes $A' = (A^{c'} \cdot (Z \cdot U^{-1} \cdot S^{-v''})^{s_e}) \bmod n$.
- qq) The assistant signer verifies that $c' = H(n \parallel Z \parallel S \parallel U \parallel v'' \parallel A \parallel A' \parallel n_H)$.
- rr) The assistant signer forward v'' to the principal signer.
- ss) The principal signer sets $v = v' + v''$ and stores (f_0, f_1, v) while the assistant signer stores (A, e) .
- tt) The group member signature key of the signer is (f_0, f_1, A, e, v) , in which (f_0, f_1, v) is the group member private key and (A, e) is the group membership credential.

6.3.3 Signature process

On input of a group public key $(n, g', g, h, S, Z, R_0, R_1, \gamma, \Gamma, \rho)$, a group member signature key (f_0, f_1, A, e, v) , a linking base bsn , a nonce $n_V \in \{0, 1\}^{l_H}$, and a message $m \in \{0, 1\}^*$ to be signed, the signature process takes the following steps below. The linking base is either a special symbol $\perp$ or an arbitrary string used for the linking capability. It is chosen either by the signer or the verifier, or pre-negotiated by both them.

NOTE 1 The nonce n_V is usually chosen by the verifier.

NOTE 2 Alternative way to handle n_V is to include n_V as part of the message m .

- a) The principal signer has the group member private key (f_0, f_1, v) while the assistant signer has (A, e) .
- b) If $bsn = \perp$, the assistant signer chooses a random integer t from $[0, \rho - 1]$ and computes $L = (f_0 + f_1 \cdot t) \bmod n$.

- c) If $bsn \neq \perp$, the assistant signer computes $J = (H_r(1 \parallel bsn))^{(\Gamma-1)/\rho} \bmod \Gamma$.
- d) The assistant signer sends J to the principal signer.
- e) The principal signer verifies that $(J)^\rho \equiv 1 \bmod \Gamma$.
- f) The principal signer computes $f = f_0 + f_1 \cdot 2^{lf}$.
- g) The principal signer computes $K = (J)^f \bmod \Gamma$.
- h) The principal signer randomly picks an integer r_v of $(l_v + l_\phi + l_H)$ -bit.
- i) The principal signer randomly picks two integers r_0, r_1 of $(l_f + l_\phi + l_H)$ -bit.
- j) The principal signer computes $T_t' = (R_0 r_0 \cdot R_1 r_1 \cdot S^{r_v}) \bmod n$.
- k) The principal signer computes $r_f' = (r_0 + r_1 \cdot 2^{lf}) \bmod \rho$.
- l) The principal signer computes $K' = (J)^{r_f'} \bmod \Gamma$.
- m) The principal signer sends K, T_t', K' to the assistant signer.
- n) The assistant signer randomly picks an integer w of $(l_n + l_\phi)$ -bit.
- o) The assistant signer randomly picks an integer r_e of $(l_e' + l_\phi + l_H)$ -bit.
- p) The assistant signer randomly picks an integer r_{v^*} of $(l_e + l_n + 2l_\phi + l_H + 1)$ -bit.
- q) The assistant signer computes $T = (A \cdot S^w) \bmod n$.
- r) The assistant signer computes $T' = (T_t' \cdot T^e \cdot S^{r_v'}) \bmod n$.
- s) The assistant signer computes $c_h = H(n \parallel R_0 \parallel R_1 \parallel S \parallel Z \parallel \gamma \parallel \Gamma \parallel \rho \parallel J \parallel T \parallel K \parallel T' \parallel K' \parallel n_V)$.
- t) The assistant signer sends c_h to the principal signer.
- u) The principal signer chooses a random nonce $n_T \in \{0, 1\}^{l_\phi}$.
- v) The principal signer computes $c = H(H(c_h \parallel n_T) \parallel m)$.
- w) The principal signer computes $s_v = r_v + c \cdot v, s_0 = r_0 + c \cdot f_0, s_1 = r_1 + c \cdot f_1$.
- x) The principal signer sends (c, n_T, s_v, s_0, s_1) to the assistant signer.
- y) The assistant signer computes $s_e = r_e + c \cdot (e - 2^{le-1})$.
- z) The assistant signer computes $s_{v'} = s_v + r_{v^*} - c \cdot w \cdot e$.
- aa) The assistant signer outputs the group signature $\sigma = (J, K, T, c, n_T, s_{v'}, s_0, s_1, s_e)$.

6.3.4 Verification process

On input of a message m , a linking base bsn , a nonce $n_V \in \{0, 1\}^{l_H}$, a group signature $(J, K, T, c, n_T, s_{v'}, s_0, s_1, s_e)$, a group public key $(n, g', g, h, S, Z, R_0, R_1, \gamma, \Gamma, \rho)$, the verification process takes the following steps:

- a) Verify that $(J)^\rho \equiv 1 \bmod \Gamma$ and $(K)^\rho \equiv 1 \bmod \Gamma$.
- b) Verify that s_0 and s_1 are at most $(l_f + l_\phi + l_H + 1)$ -bit integers.
- c) Verify that s_e is an at most $(l_e' + l_\phi + l_H + 1)$ -bit integer.
- d) Compute $t_1 = s_e + c \cdot 2^{le-1}$.
- e) Compute $t_2 = s_0 + s_1 \cdot 2^{lf}$.

- f) Compute $T' = (Z^{-c} \cdot T^{t1} \cdot R_0^{s0} \cdot R_1^{s1} \cdot S^{sv}) \bmod n$.
- g) Compute $K' = (K^{-c} \cdot J^{t2}) \bmod \Gamma$.
- h) Verify that $c = H(H(H(n \parallel R_0 \parallel R_1 \parallel S \parallel Z \parallel \gamma \parallel \Gamma \parallel \rho \parallel J \parallel T \parallel K \parallel T' \parallel K' \parallel n_V) \parallel n_T) \parallel m)$.
- i) If J is derived from linking base bsn , verify that $J = (H_\Gamma(1 \parallel bsn))^{(\Gamma-1)/\rho} \bmod \Gamma$.
- j) Optionally, call the revocation checking process.
- k) If any of the above verifications fails, output 0 (invalid), otherwise, output 1 (valid).

NOTE The revocation check in the verification process could be performed in the first step instead of the last step of the process.

6.3.5 Linking process

Given two signatures $\sigma = (J, K, T, c, n_T, s_V, s_0, s_1, s_e)$ and $\sigma' = (J', K', T', c', n_T', s_V', s_0', s_1', s_e')$, the linking process takes the following step:

- a) If $J = J'$ and $K = K'$, output 1 (linked), otherwise, output 0 (not linked).

NOTE If the linking process outputs 0 because of $J \neq J'$, it means that the linking process cannot determine whether two signatures were created by the same group member.

6.3.6 Revocation process

Details of the revocation process in this mechanism are surveyed in [10]. There are two types of revocation (private key revocation and verifier blacklist revocation) supported in this mechanism. Private key revocation can be either a global revocation or a local revocation. Verifier blacklist revocation is a local revocation.

Private key revocation:

- If a group member signature key (f_0, f_1, A, e, v) is compromised, the group membership issuer or a verifier computes $f = f_0 + f_1 \cdot 2^l$, and puts f into a revocation list RL of this type.
- Given a signature $\sigma = (J, K, T, c, n_T, s_V, s_0, s_1, s_e)$ and a revocation list RL of this type, a verifier can check revocation of this signature as follows: For each $f' \in \text{RL}$, verify $K \neq (J)^{f'} \bmod \Gamma$. If any of the verification fails, output 0 (revoked), otherwise, output 1 (valid).

NOTE The private key revocation works only if the group membership issuer or the verifier has learned the group member signature keys of the compromised group members.

Verifier blacklist revocation:

- If signatures were computed using a linking base bsn , and a verifier can build its own revocation list RL corresponding to bsn . If the verifier wants to blacklist the signer of a signature $\sigma = (J, K, T, c, n_T, s_V, s_0, s_1, s_e)$, she put K into revocation RL of this type.
- Given a signature $\sigma = (J, K, T, c, n_T, s_V, s_0, s_1, s_e)$ and a revocation list RL of this type, a verifier can check revocation of this signature as follows: For each $K' \in \text{RL}$, verify $K \neq K'$. If any of the verification fails, output 0 (revoked), otherwise, output 1 (valid).

NOTE In order to use verifier blacklist revocation in this mechanism, a signer is required to use a specific linking base for each verifier. The value of the linking base could, for example, be chosen by the verifier or agreed in advance by the signer and verifier.

6.4 Mechanism 3

6.4.1 Symbols

The following symbols apply in the specification of this mechanism.

- t : a security parameter.
- $Q_1, Q_2, A, F, R, J, K, J', K', T, R_1, R_2, R_3$: elements of G_1 .
- W : elements of G_2 .
- T_1, T_2, T_3, T_4, R_2 : elements of G_T .
- $y, f, f', x, r, c, c_h, a, b, r_f, r_x, r_a, r_b, s_f, s_x, s_a, s_b, u, v, r_u, r_v, s_u, s_v$: integers in Z_p .
- n_I, n_V, n_T : t -bit integers.
- H_1 : a hash function that outputs elements in Z_p .
- H_2 : a hash functions that output elements in G_1 .

6.4.2 Key generation process

The key generation process takes the following steps by the group membership issuer:

- a) Choose an asymmetric bilinear group pair (G_1, G_2) of large prime order p and an associated pairing function $e: G_1 \times G_2 \rightarrow G_T$.
- b) Choose a random generator P_1 of G_1 .
- c) Choose a random generator P_2 of G_2 .
- d) Choose two hash functions $H_1: \{0, 1\}^* \rightarrow Z_p$ and $H_2: \{0, 1\}^* \rightarrow G_1$. An example of how to construct such hash functions is provided in [Annex B](#).
- e) Choose random elements Q_1, Q_2 from G_1 .
- f) Choose a random integer y from Z_p^* and computes $W = [y]P_2$.
- g) Compute $T_1 = e(P_1, P_2)$, $T_2 = e(Q_1, P_2)$, $T_3 = e(Q_2, P_2)$, and $T_4 = e(Q_2, W)$.
- h) Output the following:
 - group public parameter = $(G_1, G_2, G_T, p, e, P_1, P_2, H_1, H_2)$,
 - group public key = $(Q_1, Q_2, W, T_1, T_2, T_3, T_4)$,
 - group membership issuing key = y .

NOTE 1 T_1, T_2, T_3 , and T_4 are optional in group public key, as they can be computed from P_1, P_2, Q_1, Q_2 , and W by the signers and verifiers.

NOTE 2 Examples of recommended parameters are provided in Annex [C.2](#).

The group membership issuing process requires a secure and authentic channel between the signer and the group membership issuer. How to establish such a channel is out scope of this mechanism.

For each signer, the group membership issuer generates a group member signature key as follows:

- a) Choose two random integers f, x from Z_p^* or derive them from a secret seed value.
- b) Compute $A = [1/(x+y)](P_1 + [f]Q_1)$.

- c) Output the group member signature key as (f, A, x) .

NOTE If the group member signature keys are generated using the above method, this mechanism does not provide unlinkability with respect to the group membership issuer, unless the group membership issuer deletes all the group member signature keys after provisioning them to the signers.

Alternatively, each signer can run a group membership issuing process with the group membership issuer to obtain a group member signature key:

- a) The group membership issuer chooses a nonce $n_I \in \{0, 1\}^t$.
- b) The group membership issuer sends n_I to the signer.
- c) The signer chooses at random a group member private key f from Z_p^* or derives f from its secret seed value.
- d) The signer chooses at random integer r from Z_p^* .
- e) The signer computes $F = [f]Q_1$ and $R = [r]Q_1$.
- f) The signer computes $c = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel F \parallel R \parallel n_I)$.
- g) The signer computes $s = (r + c \cdot f) \bmod p$.
- h) The signer sends (F, c, s) to the group membership issuer.
- i) The group membership issuer computes $R = [s]Q_1 - [c]F$.
- j) The group membership issuer verifies that $c = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel F \parallel R \parallel n_I)$.
- k) The group membership issuer chooses at random integer x from Z_p^* .
- l) The group membership issuer computes $A = [1/(x+y)](P_1 + F)$.
- m) The group membership issuer sets (A, x) as the signer's group membership credential and sends it to the signer.
- n) The signer verifies the credential by checking that $e(A, W + [x]P_2) = e(P_1 + F, P_2)$.
- o) The group member signature key for the signer is (f, A, x) .

6.4.3 Signature process

On input of a group public key $(Q_1, Q_2, W, T_1, T_2, T_3, T_4)$, a group member signature key (f, A, x) , a linking base bsn , and a message $m \in \{0, 1\}^*$ to be signed, the signature process takes the following steps. The linking base is either a special symbol $\perp$ or an arbitrary string used for the linking capability.

- a) If $bsn = \perp$, choose a random element J from G_1 , otherwise, compute $J = H_2(bsn)$.
- b) Compute $K = [f]J$.
- c) Choose a random integer a from Z_p^* and compute $b = (a \cdot x) \bmod p$.
- d) Compute $T = A + [a]Q_2$.
- e) Randomly pick four random integers r_f, r_x, r_a, r_b from Z_p^* .
- f) Compute $R_1 = [r_f]J$.
- g) Compute $R_2 = e(A, P_2)^{-r_x} \cdot T_2^{r_f} \cdot T_3^{r_b} \cdot a \cdot r_x \cdot T_4^{r_a}$.

NOTE $e(A, P_2)$ can be pre-computed by the group member and re-used in each signature process.

- h) Compute $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2) \parallel m)$.

- i) Compute $s_f = (r_f + c \cdot f) \bmod p$, $s_x = (r_x + c \cdot x) \bmod p$.
- j) Compute $s_a = (r_a + c \cdot a) \bmod p$, $s_b = (r_b + c \cdot b) \bmod p$.
- k) Output the anonymous digital signature $\sigma = (J, K, T, c, s_f, s_x, s_a, s_b)$.

The signature process can be jointly performed by a principal signer and an assistant signer as follows.

- a) The principal signer has the group member private key f while the assistant signer has (A, x) .
- b) If $bsn = \perp$, the principal signer chooses a random element J from G_1 , otherwise, computes $J = H_2(bsn)$.
- c) The principal signer computes $K = [f]J$.
- d) The principal signer chooses a random integer r_f from Z_p^* .
- e) The principal signer computes $R_1 = [r_f]J$ and $R_{2t} = [r_f]Q_1$.
- f) The principal signer sends (J, K, R_1, R_{2t}) to the assistant signer.
- g) The assistant signer chooses a random integer a from Z_p^* and computes $b = (a \cdot x) \bmod p$.
- h) The assistant signer computes $T = A + [a]Q_2$.
- i) The assistant signer randomly picks three integers r_x, r_a, r_b from Z_p^* .
- j) The assistant signer computes $R_2 = e(R_{2t} - [r_x]T + [r_b]Q_2, P_2) \cdot T_4^{ra}$.
- k) The assistant signer computes $c_h = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2)$.
- l) The assistant signer sends c_h to the principal signer.
- m) The principal signer chooses a nonce $n_T \in \{0, 1\}^t$.
- n) The principal signer computes $c = H_1(c_h \parallel n_T \parallel m)$.
- o) The principal signer computes $s_f = (r_f + c \cdot f) \bmod p$.
- p) The principal signer sends (c, n_T, s_f) to the assistant signer.
- q) The assistant signer computes $s_x = (r_x + c \cdot x) \bmod p$, $s_a = (r_a + c \cdot a) \bmod p$, $s_b = (r_b + c \cdot b) \bmod p$.
- r) The assistant signer outputs the anonymous digital signature $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$.

NOTE 1 The nonce n_T chosen by the principal signer is optional and can be omitted.

NOTE 2 The signature process may include a verifier's nonce n_V as an optional input. If n_V is included as input, step h) of the signature process computes $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2 \parallel n_V) \parallel m)$ and step k) of the joint signature process computes $c_h = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2 \parallel n_V)$.

6.4.4 Verification process

On input of a message m , a linking base bsn , a signature $(J, K, T, c, s_f, s_x, s_a, s_b)$, a group public key $(Q_1, Q_2, W, T_1, T_2, T_3, T_4)$, the verification process takes the following steps:

- a) Verify that J, K, T are elements in G_1 .
- b) Verify that s_f, s_x, s_a, s_b are integers in Z_p .
- c) If $bsn \neq \perp$, verify that $J = H_2(bsn)$.
- d) Compute $R_1 = [s_f]J - [c]K$.
- e) Compute $R_2 = e(T, [-s_x]P_2 - [c]W) \cdot T_1^c \cdot T_2^{s_f} \cdot T_3^{s_b} \cdot T_4^{s_a}$.

- f) Verify that $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2) \parallel m)$.
- g) Optionally, call the revocation checking process.
- h) If any of the above verifications fails, output 0 (invalid), otherwise, output 1 (valid).

NOTE If nonce n_T is included in the signature, then step f) of the verification process verifies that $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2) \parallel n_T \parallel m)$.

6.4.5 Linking process

Given two signatures $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ and $\sigma' = (J', K', T', c', n_{T'}, s_{f'}, s_{x'}, s_{a'}, s_{b'})$, the linking process takes the following step:

- a) If $J = J'$ and $K = K'$, output 1 (linked), otherwise, output 0 (not linked).

NOTE If the linking process outputs 0 because of $J \neq J'$, it means that the linking process cannot determine whether two signatures were created by the same group member.

6.4.6 Revocation process

Details of the revocation process in this mechanism are surveyed in [10]. There are three types of revocation (private key revocation, verifier blacklist revocation, and signature revocation) supported in this mechanism. Private key revocation and signature revocation can be either global revocation or local revocation. Verifier blacklist revocation is a local revocation.

Private key revocation:

- If a group member signature key (f, A, x) is compromised, the group membership issuer or a verifier puts f into a revocation list RL of this type.
- Given a signature $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ and a revocation list RL of this type, a verifier can check revocation of this signature as follows: For each $f' \in \text{RL}$, verify $K \neq [f']J$. If any of the verification fails, output 0 (revoked), otherwise, output 1 (valid).

Verifier blacklist revocation:

- If signatures were computed using a linking base bsn , and a verifier can build its own revocation list RL corresponding to bsn . If the verifier wants to blacklist the signer of a signature $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$, she put K into revocation RL of this type.
- Given a signature $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ and a revocation list RL of this type, a verifier can check revocation of this signature as follows: For each $K' \in \text{RL}$, verify $K \neq K'$. If any of the verification fails, output 0 (revoked), otherwise, output 1 (valid).

NOTE In order to use verifier blacklist revocation in this mechanism, a signer is required to use a specific linking base for each verifier. The value of the linking base could, for example, be chosen by the verifier or agreed in advance by the signer and verifier.

Signature revocation:

- If the group membership issuer or a verifier determines that a signature $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ was created by a corrupted signer but has not obtained the corresponding group member signature key, the issuer or the verifier places (J, K) pair of the signature into a revocation list RL of this type.
- In order to perform revocation check, the signer needs to use zero-knowledge proof to prove that she did not create any of the (J', K') in RL before. More specifically, let $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ be the signature that the signer just created, for each (J', K') pair in RL, the signer proves to the verifier in zero-knowledge that $[f]J = K$ and $[f]J' \neq K'$ as follows:
 - a) The signer chooses a random integer u from Z_p^* .

- b) The signer computes $v = (-f \cdot u) \bmod p$.
- c) The signer computes $T = [u]K' + [v]J'$. If $T = O_E$, non-revoked proof fails.
- d) The signer chooses two random integer r_u, r_v from Z_p^* .
- e) The signer computes $R_1 = [r_u]K + [r_v]J$ and $R_3 = [r_u]K' + [r_v]J'$.
- f) The signer computes $c = H_1(p \parallel P_1 \parallel J \parallel K \parallel J' \parallel K' \parallel T \parallel R_1 \parallel R_3 \parallel m)$.
- g) The signer computes $s_u = (r_u + c \cdot u) \bmod p$ and $s_v = (r_v + c \cdot v) \bmod p$.
- h) The signer sends (T, c, s_u, s_v) as the non-revoked proof to the verifier.
- i) The verifier verifies that $T \in G_1$ and $s_u, s_v \in Z_p$.
- j) The verifier verifies that $T \neq O_E$.
- k) The verifier computes $R_1 = [s_u]K + [s_v]J$ and $R_3 = [s_u]K' + [s_v]J' - [c]T$.
- l) The verifier verifies that $c = H_1(p \parallel P_1 \parallel J \parallel K \parallel J' \parallel K' \parallel T \parallel R_1 \parallel R_3 \parallel m)$.
- m) If any of the verification steps fails, the verifier rejects the non-revoked proof, otherwise, accepts the proof.

NOTE 1 The signature revocation requires an interactive process between the signer and verifier.

NOTE 2 To preserve anonymity, it is recommended to have a trusted entity for updating the signature revocation list. If a malicious entity controls the signature revocation list, the anonymity of the signer can be reduced.

6.5 Mechanism 4

6.5.1 Symbols

The following symbols apply in the specification of this mechanism.

- t : a security parameter.
- $Q_2, U', A, B, C, D, R, S, T, W, J, K, R_1, R_2, C'$: elements of G_1 .
- X, Y, X' : elements of G_2 .
- $x, y, f, u, v, w, v', l, c, r, h, s, x', \beta$: integers in Z_p .
- n_l, n_v, n_T : t -bit integers.
- H_1 : a hash function that outputs elements in G_1 .
- H_2, H_3, H_4 : hash functions that output elements in Z_p .

6.5.2 Key generation process

The key generation process has two parts: setup process and group membership issuing process. The setup process is executed by the group membership issuer to create the group public parameter, group public key, and group membership issuing key. The group membership issuing process is an interactive protocol running between the group membership issuer and a group member to create a unique group member signature key for the group member.

The setup process takes the following steps by the group membership issuer:

- a) Choose t as a security parameter.

- b) Choose an asymmetric bilinear group pair (G_1, G_2) of large prime order p and an associated pairing function $e: G_1 \times G_2 \rightarrow G_T$.
- c) Choose a random generator P_1 of G_1 .
- d) Choose a random generator P_2 of G_2 .
- e) Choose four hash functions $H_1: \{0, 1\}^* \rightarrow G_1$, $H_2: \{0, 1\}^* \rightarrow Z_p$, $H_3: \{0, 1\}^* \rightarrow Z_p$, $H_4: \{0, 1\}^* \rightarrow Z_p$. An example of how to construct such hash functions is provided in [Annex B](#).
- f) Choose two random integers x, y in Z_p .
- g) Compute $X = [x]P_2$ and $Y = [y]P_2$.
- h) Output the following:
 - group public parameter = $(G_1, G_2, G_T, e, P_1, P_2, p, H_1, H_2, H_3, H_4)$,
 - group public key = (X, Y) ,
 - group membership issuing key = (x, y) .

NOTE Examples of recommended parameters are provided in Annex [C.2](#).

The group membership issuing process requires a secure and authentic channel between the principal signer and the group membership issuer. How to establish such a channel is out scope of this mechanism. The group membership issuing process includes the following steps:

- a) The group membership issuer chooses a nonce $n_I \in \{0, 1\}^t$.
- b) The group membership issuer sends n_I to the principal signer.
- c) The principal signer chooses a group member private key f randomly from Z_p or derives f from its secret seed value.
- d) The principal signer computes $Q_2 = [f]P_1$.
- e) The principal signer chooses u from Z_p and computes $U = [u]P_1$.
- f) The principal signer computes $v = H_2(P_1 || Q_2 || U || X || Y || n_I)$.
- g) The principal signer computes $w = (u + v \cdot f) \bmod p$.
- h) The principal signer sends (Q_2, v, w) to the group membership issuer.
- i) The group membership issuer computes $U' = [w]P_1 - [v]Q_2$.
- j) The group membership issuer computes $v' = H_2(P_1 || Q_2 || U' || X || Y || n_I)$.
- k) The group membership issuer verifies $v = v'$. If the verification fails, abort the group membership issuing process.
- l) The group membership issuer chooses a random integer r from Z_p .
- m) The group membership issuer computes $A = [r]P_1$, $B = [y]A$, $C = [x]A + [rxy]Q_2$.
- n) The group membership issuer sets (A, B, C) as the signer's group membership credential and sends the credential to the principal signer.
- o) The principal signer computes $D = [f]B$.
- p) The principal signer sends (A, B, C, D) to the assistant signer.
- q) The assistant signer verifies $e(A, Y) = e(B, P_2)$.

- r) The assistant signer verifies $e(A + D, X) = e(C, P_2)$.
- s) If the verification fails, the assistant signer aborts.
- t) The group member signature key for the signer is (f, A, B, C) .

6.5.3 Signature process

On input of a group public key (X, Y) , a group member signature key (f, A, B, C) , a linking base bsn , a nonce $n_V \in \{0, 1\}^t$, and a message $m \in \{0, 1\}^*$ to be signed, the signature process takes the following steps. The linking base is either a special symbol $\perp$ or an arbitrary string used for the linking capability.

NOTE 1 The nonce n_V is usually chosen by the verifier.

NOTE 2 Alternative way to handle n_V is to include n_V as part of the message m .

- a) The principal signer has the group member private key f while the assistant signer has (A, B, C, D) .
- b) If $bsn = \perp$, the assistant signer chooses a random J from G_1 , otherwise, computes $J = H_1(bsn)$.
- c) The assistant signer chooses a random integer l from Z_p .
- d) The assistant signer computes $R = [l]A$, $S = [l]B$, $T = [l]C$, and $W = [l]D$.
- e) The assistant signer computes $c = H_3(R \parallel S \parallel T \parallel W \parallel n_V)$.
- f) The assistant signer sends (c, J, S, m, bsn) to the principal signer.
- g) The principal signer computes $K = [f]J$.
- h) The principal signer chooses $n_T \in \{0, 1\}^t$.
- i) The principal signer chooses a random integer r from Z_p .
- j) The principal signer computes $R_1 = [r]J$ and $R_2 = [r]S$.
- k) The principal signer computes $h = H_4(c \parallel m \parallel J \parallel K \parallel bsn \parallel R_1 \parallel R_2 \parallel n_T)$.
- l) The principal signer computes $s = (r + h \cdot f) \bmod p$.
- m) The principal signer sends (K, h, s, n_T) to the assistant signer.
- n) The assistant signer outputs the anonymous signature $\sigma = (R, S, T, W, J, K, h, s, n_V, n_T)$.

6.5.4 Verification process

On input of a message m , a linking base bsn , a nonce $n_V \in \{0, 1\}^t$, a signature $(R, S, T, W, J, K, h, s, n_V, n_T)$, a group public key (X, Y) , the verification process takes the following steps:

- a) If $bsn \neq \perp$, verify that $J = H_1(bsn)$.
- b) Verify that $e(R, Y) = e(S, P_2)$ and $e(R + W, X) = e(T, P_2)$.
- c) Compute $R_1 = [s]J - [h]K$.
- d) Compute $R_2 = [s]S - [h]W$.
- e) Verify that $h = H_4(H_3(R \parallel S \parallel T \parallel W \parallel n_V) \parallel m \parallel J \parallel K \parallel bsn \parallel R_1 \parallel R_2 \parallel n_T)$.
- f) Optionally, call the revocation checking process.
- g) Verify that $R \neq O_E$.
- h) If any of the above verifications fails, output 0 (invalid), otherwise, output 1 (valid).

6.5.5 Linking process

Given two signatures $\sigma = (R, S, T, W, J, K, h, s, n_V, n_T)$ and $\sigma' = (R', S', T', W', J', K', h', s', n_V', n_T')$, the linking process takes the following step:

- a) If $J = J'$ and $K = K'$, output 1 (linked), otherwise, output 0 (not linked).

NOTE If the linking process outputs 0 because of $J \neq J'$, it means that the linking process cannot determine whether two signatures were created by the same group member.

6.5.6 Revocation process

Details of the revocation process in this mechanism are surveyed in [10]. There are four types of revocation (private key revocation, verifier blacklist revocation, signature revocation, and credential update) supported in this mechanism. The first three revocations are the same in 6.4.6. The credential update revocation process is specified below. Private key revocation and signature revocation can be either global revocation or local revocation. Verifier blacklist revocation is a local revocation. Credential update is a global revocation.

Updating group public key process:

- a) The group membership issuer chooses randomly x' in Z_p .
- b) The group membership issuer computes $X' = [x']P_2$.
- c) The new group public key is (X', Y) .
- d) The new group membership issuing key is (x', y) .

Updating membership credential process:

- a) The group membership issuer computes $\beta = x'/x \bmod p$.
- b) For each legitimate member with group membership credential (A, B, C) ,
 - 1) The issuer computes $C' = [\beta]C$ and sends C' to the member.
 - 2) The member updates its credential as (A, B, C') .

6.6 Mechanism 8

6.6.1 Symbols

The following symbols apply in the specification of this mechanism.

- τ : a security parameter.
- $P_1, Q_1, X_1, Y_1, X'_1, \tilde{X}_1, C_1, D, D', T_1, T_2, K_1, K_2, K, K'_1, K'_2, K', J, T'_1, T'_2, R, R', T, T', R'', T''$: elements of G_1 .
- $P_2, X_2, Y_2, X'_2, \tilde{X}_2$: elements of G_2 .
- $x, y, z, x', z', c_k, s_x, s_z, c'_k, s_1, u, v, w, v', r, s_2, k_r, k_x, k_z, c, z_r, z_x, z_z, c', s, l, k_s, c_m, \circ, c'_m$: integers in Z_p .

- n_I : an integer of size τ -bit.
- H_1 : a hash function that outputs elements in G_1 .
- H_2, H_3 : hash functions that output elements in Z_p .

6.6.2 Key generation process

The key generation process has two parts: setup process and group membership issuing process. The setup process is executed by the group membership issuer to create the group public parameter, group public key, and group membership issuing key. The group membership issuing process is an interactive protocol running between the group membership issuer and a group member to create a unique group member signature key for the group member.

The setup process takes the following steps by the group membership issuer:

- a) Choose τ as a security parameter.
- b) Choose a bilinear group pair (G_1, G_2) of large prime order p , such that no efficiently computable homomorphism is known between G_1 and G_2 , in either direction, and an associated pairing function $e: G_1 \times G_2 \rightarrow G_T$.
- c) Choose two random independent generators P_1 and Q_1 of G_1 and provide additional information, denoted by π_{Gen} , that serve to demonstrate that these two generators were indeed chosen independently, that is without a potentially exploitable relationship between them (such as $Q_1 = [s]P_1$ for an integer s chosen by the group membership issuer). An example of how to verifiably select independent generators and to verify, using π_{Gen} , the correct generation of these generators, is given in Annex G.
- d) Choose a random generator P_2 of G_2 .
- e) Choose three hash functions $H_1: \{0,1\}^* \rightarrow G_1$, $H_2: \{0,1\}^* \rightarrow Z_p$ and $H_3: \{0,1\}^* \rightarrow Z_p$. An example of how to construct such hash functions is provided in Annex B.
- f) Choose three random integers x , y and z in Z_p .
- g) Compute $X_1 = [z]P_1 + [x]Q_1$, $Y_1 = [y]P_1$, $X_2 = [x]P_2$ and $Y_2 = [y]P_2$.
- h) Choose two random integers x' and z' in Z_p .
- i) Compute $X'_1 = [z']P_1 + [x']Q_1$ and $X'_2 = [x']P_2$.
- j) Compute $c_k = H_2(P_1 || Q_1 || P_2 || X_1 || Y_1 || X_2 || Y_2 || X'_1 || X'_2)$.
- k) Compute $s_x = (x' + c_k \times x) \bmod p$ and $s_z = (z' + c_k \times z) \bmod p$.
- l) Set $\pi_{\text{Val}} = (c_k, s_x, s_z)$ as a proof that the second component of the representation of X_1 in the base P_1 and Q_1 is equal to the discrete logarithm of X_2 in the base P_2 .
- m) Output the following:
 - group public parameter = $(G_1, G_2, G_T, e, P_1, Q_1, P_2, p, H_1, H_2, H_3)$,
 - group public key = $(X_1, Y_1, X_2, Y_2, \pi_{\text{Gen}}, \pi_{\text{Val}})$,
 - group membership issuing key = (x, y, z) .

NOTE 1 Examples of recommended parameters are provided in C.2.

Each entity involved in this anonymous signature mechanism should verify the validity of the group public key before using it. The group public key validity verification process includes the following steps:

- a) Verify that P_1 and Q_1 were generated independently using π_{Gen} .
- b) Verify the validity of the proof π_{Val} :
 - 1) Compute $\tilde{X}_1 = [s_z]P_1 + [s_x]Q_1 - [c_k]X_1$ and $\tilde{X}_2 = [s_x]P_2 - [c_k]X_2$.
 - 2) Compute $c'_k = H_2(P_1 \parallel Q_1 \parallel P_2 \parallel X_1 \parallel Y_1 \parallel X_2 \parallel Y_2 \parallel \tilde{X}_1 \parallel \tilde{X}_2)$.
 - 3) Verify that $c'_k = c_k$.
- c) Verify that $e(Y_1, P_2) = e(P_1, Y_2)$.
- d) If any of the above verifications fails, output 0 (invalid), otherwise output 1 (valid).

The group membership issuing process requires a secure and authentic channel between the group member and the group membership issuer. How to establish such a channel is out scope of this mechanism. The group membership issuing process includes the following steps:

- a) The group membership issuer chooses a nonce $n_I \in \{0, 1\}^\tau$.
- b) The group membership issuer sends n_I to the member.
- c) The member chooses a random integer s_1 from Z_p .
- d) The member computes $C_1 = [s_1]Y_1$.
- e) The member chooses a random integer u from Z_p .
- f) The member computes $D = [u]Y_1$.
- g) The member computes $v = H_2(P_1 \parallel Q_1 \parallel P_2 \parallel X_1 \parallel Y_1 \parallel X_2 \parallel Y_2 \parallel C_1 \parallel D \parallel n_I)$.
- h) The member computes $w = (u + v \times s_1) \bmod p$.
- i) The member sends (C_1, v, w) to the group membership issuer.
- j) The group membership issuer computes $D' = [w]Y_1 - [v]C_1$.
- k) The group membership issuer computes $v' = H_2(P_1 \parallel Q_1 \parallel P_2 \parallel X_1 \parallel Y_1 \parallel X_2 \parallel Y_2 \parallel C_1 \parallel D' \parallel n_I)$.
- l) The group membership issuer verifies $v = v'$. If the verification fails, abort the group membership issuing process.
- m) The group membership issuer selects five random integers r, s_2, k_r, k_x and k_z from Z_p .
- n) The group membership issuer computes $T_1 = [r]P_1$ and $T_2 = [x]T_1 + [r]C_1 + [r \times s_2]Y_1$.
- o) The group membership issuer computes $K_1 = [k_r]P_1$, $K_2 = [k_x]T_1 + [k_r](C_1 + [s_2]Y_1)$ and $K = [k_z]P_1 + [k_x]Q_1$.
- p) The group membership issuer computes $c = H_2(P_1 \parallel Q_1 \parallel P_2 \parallel X_1 \parallel Y_1 \parallel X_2 \parallel Y_2 \parallel C_1 \parallel s_2 \parallel K_1 \parallel K_2 \parallel K)$.
- q) The group membership issuer computes $z_r = (k_r + c \times r) \bmod p$, $z_x = (k_x + c \times x) \bmod p$ and $z_z = (k_z + c \times z) \bmod p$.
- r) The group membership issuer sets (T_1, T_2) as the member's group membership credential and sends $(T_1, T_2, s_2, c, z_r, z_x, z_z)$ to the member.

- s) The member computes $K'_1 = [z_r]P_1 - [c]T_1$, $K'_2 = [z_x]T_1 + [z_r](C_1 + [s_2]Y_1) - [c]T_2$ and $K' = [z_z]P_1 + [z_x]Q_1 - [c]X_1$.
- t) The member computes $c' = H_2(P_1 \parallel Q_1 \parallel P_2 \parallel X_1 \parallel Y_1 \parallel X_2 \parallel Y_2 \parallel C_1 \parallel s_2 \parallel K'_1 \parallel K'_2 \parallel K)$.
- u) The member verifies $c = c'$. If the verification fails, the member aborts.
- v) The member computes $s = (s_1 + s_2) \bmod p$.
- w) The group member signature key for the member is (s, T_1, T_2) .

NOTE 2 The group membership issuer can use the same value s_2 (for example $s_2 = 0 \bmod p$) for several executions of the group membership issuing process. In this case, the security of Mechanism 8 relies on the Pointcheval-Sanders (PS) assumption^[24], instead of the q-MSDH assumption^[25] if the group membership issuer uses a fresh random value s_2 for each new session of the group membership issuing process.

6.6.3 Signature process

On input of a group member signature key (s, T_1, T_2) , a linking base bsn and a message $m \in \{0,1\}^*$ to be signed, the signature process takes the following steps. The linking base, denoted by bsn , is either a special symbol $\perp$ or an arbitrary string used for the linking capability.

- a) If $bsn = \perp$, the signer chooses a random J from G_1 , otherwise, computes $J = H_1(bsn)$.
- b) The signer selects two random integers l and k_s in Z_p .
- c) The signer computes $T'_1 = [l]T_1$ and $T'_2 = [l]T_2$.
- d) The signer computes $R = [s]T'_1$ and $R' = [k_s]T'_1$.
- e) The signer computes $T = [s]J$ and $T' = [k_s]J$.
- f) The signer computes $c_m = H_3(T'_1 \parallel T'_2 \parallel J \parallel T \parallel R \parallel T' \parallel R' \parallel m)$.
- g) The signer computes $\rho = (k_s + c_m \times s) \bmod p$.
- h) The signer outputs the anonymous signature $\sigma = (T'_1, T'_2, J, R, T, c_m, \rho)$.

6.6.4 Verification process

On input of a message m , a linking base bsn , a signature $(T'_1, T'_2, J, R, T, c_m, \rho)$ and a group public key (X_1, Y_1, X_2, Y_2) , the verification process takes the following steps:

- a) If $bsn \neq \perp$, verify that $J = H_1(bsn)$.
- b) Verify that $T'_1 \neq O_E$.
- c) If any of the above verifications fails, output 0 (invalid).
- d) Compute $R'' = [\rho]T'_1 - [c_m]R$.
- e) Compute $T'' = [\rho]J - [c_m]T$.
- f) Compute $c'_m = H_3(T'_1 \parallel T'_2 \parallel J \parallel T \parallel R \parallel T'' \parallel R'' \parallel m)$.

- g) Verify that $c'_m = c_m$.
- h) Verify that $e(T'_1, X_2) \times e(R, Y_2) = e(T'_2, P_2)$.
- i) Optionally, call the revocation checking process.
- j) If any of the above verifications (steps g and h) fails, output 0 (invalid). Otherwise, output 1 (valid).

6.6.5 Linking process

Given two valid signatures $\sigma = (T'_1, T'_2, J, R, T, c_m, \rho)$ and $\hat{\sigma} = (\hat{T}'_1, \hat{T}'_2, \hat{J}, \hat{R}, \hat{T}, \hat{c}_m, \hat{\rho})$, the linking process takes the following steps:

- a) If $J = \hat{J}$ and $T = \hat{T}$, output 1 (linked), otherwise, output 0 (not linked).

NOTE If the linking process outputs 0 because of $J \neq \hat{J}$, it means that the linking process cannot determine whether two signatures were created by the same group member.

6.6.6 Revocation process

Details of the revocation process in this mechanism are surveyed in Reference [10]. There are two types of revocation (private key revocation and verifier blacklist revocation) supported in this mechanism. Private key revocation can be either global revocation or local revocation. Verifier blacklist revocation is a local revocation.

Private key revocation:

- If a group member signature key (s, T_1, T_2) is compromised, the group membership issuer puts s into a revocation list RL of this type.
- Given a valid signature $\sigma = (T'_1, T'_2, J, R, T, c_m, \rho)$ computed using a linking base bsn and a revocation list RL of this type, a verifier can check revocation of this signature as follows: for each $s' \in RL$, verify $T \neq [s']J$. If any of these verifications fails, output 0 (revoked), otherwise, output 1 (valid).

NOTE The private key revocation works only if the group membership issuer or the verifier has learned the group member signature keys of the compromised group members. This revocation process allows to identify every group signature generated using this private key. If this key can be associated with a group member (e.g. by using contextual information), then no anonymity can be retained for this group member as their signatures can therefore be traced. This is a property inherent to DAA schemes. Thus, a careful assessment of the need for revocation and the consequences for the corresponding group member will be carried out before deployment.

Verifier blacklist revocation:

- If signatures were computed using a linking base bsn , a verifier can build its own revocation list RL corresponding to bsn . If the verifier wants to blacklist the signer of a valid signature $\sigma = (T'_1, T'_2, J, R, T, c_m, \rho)$, they put T into a revocation list RL of this type.
- Given a signature $\sigma = (T'_1, T'_2, J, R, T, c_m, \rho)$ computed using a linking base bsn and a revocation list RL of this type, a verifier can check revocation of this signature as follows: for each $\hat{T} \in RL$, verify $T \neq \hat{T}$. If any of these verifications fails, output 0 (revoked), otherwise, output 1 (valid).

In order to use verifier blacklist revocation in this mechanism, a signer must use a specific linking base for each verifier. The value of the linking base can, for example, be chosen by the verifier or agreed in advance by the signer and verifier. [A2](#)

7 Mechanisms with opening capability

7.1 General

A2 This clause specifies three digital signature mechanisms with opening capability. **A2** This type of digital signatures is called group signatures in the literature. In these mechanisms, there is an entity called group membership opener who is designated by the signer and who can identify the signer from a group signature. Each mechanism consists of key generation process, signature process, verification process, opening process, and revocation process.

A2 NOTE The mechanisms and associated security proofs in 7.2, 7.3 and 7.4 are based on References [17], [14] and [23] (an extended version of Reference [24]), respectively. **A2**

7.2 Mechanism 5

7.2.1 Symbols

The following symbols apply in the specification of this mechanism.

- K_n, K, K_C, K_S : security parameters.
- p_1', p_2' : prime numbers.
- p_1, p_2 : $K_n/2$ -bit prime numbers such that $p_1 = 2p_1' + 1, p_2 = 2p_2' + 1$.
- n : K_n -bit integer such that $n = p_1 p_2$.
- a_0, a_1, a_2, b, w : elements of $QR(n)$.
- G : a group which DDH assumption holds.
- q : order of G .
- g : a generator of G .
- y_1, y_2 : elements of Z_q .
- Y_1, Y_2 : elements of G .