



中华人民共和国国家标准

GB/T 38647.2—2020

信息技术 安全技术 匿名数字签名 第2部分：采用群组公钥的机制

Information technology—Security techniques—Anonymous digital signatures—
Part 2: Mechanisms using a group public key

(ISO/IEC 20008-2:2013, MOD)

2020-04-28 发布

2020-11-01 实施

国家市场监督管理总局 发布
国家标准化管理委员会

目 次

前言 I

引言 II

1 范围 1

2 规范性引用文件 1

3 术语和定义 1

4 符号 2

5 一般模型和要求 3

6 具有连接能力的机制 4

 6.1 概述 4

 6.2 机制 1 4

 6.3 机制 2 8

 6.4 机制 3 13

 6.5 机制 4 16

7 具有打开功能的机制 19

 7.1 概述 19

 7.2 机制 5 19

 7.3 机制 6 22

8 具有打开和连接功能的机制 24

 8.1 概述 24

 8.2 机制 7 24

 8.3 机制 8 28

附录 A（规范性附录） 对象标识符 33

附录 B（规范性附录） 密码杂凑函数 35

附录 C（资料性附录） 采用群组公钥的匿名签名机制的安全指南 37

附录 D（资料性附录） 撤销机制的比较 40

附录 E（资料性附录） 数值实例 43

附录 F（资料性附录） 机制 5 的正确性证明 95

参考文献 99

前 言

GB/T 38647《信息技术 安全技术 匿名数字签名》拟分为两个部分：

- 第 1 部分：总则；
- 第 2 部分：采用群组公钥的机制。

本部分为 GB/T 38647 的第 2 部分。

本部分按照 GB/T 1.1—2009 给出的规则起草。

本部分使用重新起草法修改采用 ISO/IEC 20008-2:2013《信息技术 安全技术 匿名数字签名 第 2 部分：采用群组公钥的机制》。

本部分与 ISO/IEC 20008-2:2013 的技术性差异及其原因如下：

——关于规范性引用文件，本部分做了具有技术性差异的调整，以适应我国的技术条件，调整的情况集中反映在第 2 章“规范性引用文件”中，具体调整如下：

- 删除了 ISO/IEC 18031 和 ISO/IEC 18032；
- 增加了 GB/T 32905、GB/T 32918.2—2016、GB/T 34953.2—2018 和 ISO/IEC 15946-1；
- 用修改采用国际标准的 GB/T 38647.1 代替了 ISO/IEC 20008-1。

——第 5 章采用了我国密码算法国家标准 GB/T 32905，以与我国技术水平相适应。

——第 8 章增加了机制 8(见 8.3)，该机制基于 GB/T 32918.2—2016，是与我国商用密码算法相适应的匿名数字签名技术。

——增加了与机制 8 的数学假设和安全参数选取相关的内容(见附录 C)。

——增加了与机制 8 的撤销机制相关的内容(见附录 D)。

——增加了 E.8，给出了机制 8 的数值实例(见附录 E)。

本部分由全国信息安全标准化技术委员会(SAC/TC 260)提出并归口。

本部分起草单位：西安西电捷通无线网络通信股份有限公司、无线网络安全技术国家工程实验室、中关村无线网络安全产业联盟、国家密码管理局商用密码检测中心、国家无线电监测中心检测中心、国家信息技术安全研究中心、中国通用技术研究院、中国电子技术标准化研究院、天津市电子机电产品检测中心、重庆邮电大学、北京计算机技术及应用研究所、工业和信息化部宽带无线 IP 标准工作组。

本部分主要起草人：杜志强、曹军、张国强、李琴、李志勇、李冬、赵晓荣、黄振海、李冰、陶洪波、刘科伟、颜湘、刘景莉、赵旭东、王月辉、张璐璐、吕春梅、许玉娜、傅强、龙昭华、彭潇、熊克琦、林德欣、铁满霞、吴冬宇、郑骊、高德龙、张变玲、于光明、朱正美、赵慧、黄奎刚。

引 言

匿名数字签名机制是一种特殊类型的数字签名机制,该类数字签名机制中,非授权的实体不能获得签名方的身份标识,但可验证合法的签名方产生了合法的签名。

采用群组公钥的匿名数字签名机制具有提供签名方更多信息的能力。一些签名机制拥有连接能力,其中由同一个签名方产生的两个签名是可连接的。一些签名机制拥有打开能力,其中签名可以被特殊的实体打开来揭露签名方的身份。一些签名机制既具有连接能力也具有打开能力。

对于每个机制,本部分规定了打开、连接和/或撤销过程。

本部分规定的机制使用了 GB/T 32905 中规范的抗碰撞密码杂凑函数去计算整个消息。

本文件的发布机构提请注意,声明符合本文件时,可能涉及与 8.3 相关的 CN201810207503.4、CN201810207564.0、CN201810207571.0 等专利的使用。

本文件的发布机构对于上述专利的真实性、有效性和范围无任何立场。

该专利持有人已向本文件的发布机构保证,他愿意同任何申请人在合理且无歧视的条款和条件下,就专利授权许可进行谈判。该专利持有人的声明已在本文件发布机构备案。相关信息可通过以下联系方式获得:

专利持有人:西安西电捷通无线网络通信股份有限公司
地址:西安市高新区科技二路 68 号西安软件园秦风阁 A201
联系人:冯玉晨
邮政编码:710075
电子邮件:ipri@iwncomm.com
电话:029-87607836
传真:029-87607829
网址:<http://www.iwncomm.com>

本文件的发布机构提请注意,本文件修改采用 ISO/IEC 20008-2:2013,因此,除上述声明外,韩国电子通信研究院、英特尔公司以及日本 NEC 公司针对 ISO/IEC 20008-2:2013 所作出的“专利持有人愿意基于无歧视、合理条件和条款与其他方协商许可”的声明适用于本文件。相关信息可通过以下联系方式获得:

专利持有人:Electronics and Telecommunications Research Institute
地址:161,Gajeong-dong,Yuseong-gu,Daejeon,305-700,KOREA
联系人:Hanchul Shin
电子邮件:vip123@etri.ke.kr
电话:+82-042-860-5797
传真:+82-042-860-3831
网址:<http://www.etri.re.kr>

专利持有人:Intel Corporation
地址:Intel Legal and Corporation Affairs 2200 Mission College Blvd.,RNB-150,Santa Clara,CA 95054
联系人:James Kovacs
电子邮件:Standards.Licensing@intel.com
电话:408-765-1170

传真:408-613-7292

网址:<http://www.intel.com/standards/licensing.html>

专利持有人:NEC Corporation

地址:7-1 Shiba 5-chome Minato-Ku TokyoJapan 108-8001 Japan

联系人:Yoshinobu Matsumoto

电子邮件:y-matsumoto@cp.jp.nec.com

电话:+81-3-3798-2452

传真:+81-3-3798-6394

网址:<http://www.nec.com/>

请注意除上述专利外,本文件的某些内容仍可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

信息技术 安全技术 匿名数字签名 第2部分：采用群组公钥的机制

1 范围

GB/T 38647 的本部分规定了采用群组公钥的匿名数字签名机制的一般模型和要求、具有连接能力的机制、具有打开功能的机制、具有打开和连接功能的机制。

本部分给出了：

- a) 采用群组公钥签名的匿名数字签名机制的概述；
- b) 多种提供这类匿名数字签名的机制。

对于每个机制，本部分规定了：

- a) 群组成员签名密钥和群组公钥的生成过程；
- b) 生成签名的过程；
- c) 验证签名的过程；
- d) 群组成员打开过程(可选)；
- e) 群组签名连接过程(可选)；
- f) 撤销群组签名的过程。

本部分适用于指导采用群组公钥的匿名数字签名机制的设计、实现与应用。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 32905 信息安全技术 SM3 密码杂凑算法

GB/T 32918.2—2016 信息安全技术 SM2 椭圆曲线公钥密码算法 第2部分：数字签名算法

GB/T 34953.2—2018 信息技术 安全技术 匿名实体鉴别 第2部分：基于群组公钥签名的机制(ISO/IEC 20009-2:2013, IDT)

GB/T 38647.1 信息技术 安全技术 匿名数字签名 第1部分：总则(GB/T 38647.1—2020, ISO/IEC 20008-1:2013, MOD)

ISO/IEC 10118(所有部分) 信息技术 安全技术 杂凑函数(Information technology—Security techniques—Hash-functions)

ISO/IEC 15946-1 信息技术 安全技术 基于椭圆曲线的密码技术 第1部分：通用要求(Information technology—Security techniques—Cryptographic techniques based on elliptic curves—Part 1: General)

ISO/IEC 15946-5 信息技术 安全技术 基于椭圆曲线的密码技术 第5部分：椭圆曲线生成(Information technology—Security techniques—Cryptographic techniques based on elliptic curves—Part 5: Elliptic curve generation)

3 术语和定义

GB/T 38647.1 界定的以及下列术语和定义适用于本文件。

3.1

辅助签名方 assistant signer

可以帮助主签名方去创建匿名签名,但不可以独立地产生匿名签名的实体。

3.2

成员列表 member-list

包含群组成员身份及其对应的群组成员证书的列表。

3.3

主签名方 principal signer

拥有群组成员签名密钥,并使用该密钥来创建匿名签名的实体。

3.4

密钥种子值 secret seed value

群组成员所知的并且用来导出群组成员私钥的保密数据。

3.5

安全参数 security parameter

决定一个机制安全强度的变量。

4 符号

下列符号适用于本文件。

bsn :连接基,既可以是一个特殊符号 $\perp$,也可以是任意字符串。

$e(\cdot)$:双线性映射函数 $e:G_1 \times G_2 \rightarrow G_T$,使得对所有 $P \in G_1, Q \in G_2$,所有正整数 a, b ,等式 $e([a]P, [b]Q) = e(P, Q)^{ab}$ 成立,该函数也称为对函数。

$\gcd(a, b)$:整数 a 和 b 的最大公因子。

G_1 :椭圆曲线上的阶为 p 的一个加法循环群。

G_2 :椭圆曲线上的阶为 p 的一个加法循环群。

G_T :阶为 p 的乘法循环群。

H :密码杂凑函数。

m :待签名消息。

n :RSA 模块,其中 $n = pq$ 。

O_E :椭圆曲线上的无穷大点。

P :素数。

P_1 : G_1 的生成元。

P_2 : G_2 的生成元。

q :素数。

$Q_1 + Q_2$:椭圆曲线点 Q_1 和 Q_2 的总和。

$QR(n)$:模 n 的二次剩余群组。

Z_n^* : Z_n 中可逆元素的乘法群。

Z_p : $[0, p-1]$ 中整数的集合。

Z_p^* : $[1, p-1]$ 中整数的集合。

$(a|p)$: a 和 p 的勒让德符号,其中 a 是一个整数, p 是一个奇素数。

$[n]P$:一个正整数和一个在椭圆曲线 E 上的点 P 之间的乘法运算, n 和 P 作为输入并产生另外椭圆曲线 E 上的点 Q ,其中 $Q = [n]P = P + P + \dots + P$,是 n 个 P 的总和。该运算满足 $[0]P = O_E$ 和 $[-n]P = [n](-P)$ 。

$[x, y]$:从 x 到 y 之间包括 x 和 y 的整数集合,如果 x, y 是整数,则满足 $x \leq y$ 。

$\parallel : X \parallel Y$ 表示数据项 X 和 Y 以特定的顺序级联的结果。将两个或多个数据项连接的结果作为本部分规定的机制之一进行签名时,应将该结果组合起来,使其可以唯一的分解为其组成数据项,确保不存在歧义的可能,这种属性可根据应用以各种不同的方式实现。例如,它可以:a)在整个机制被使用时确保子字符串的固定长度,或 b)确保使用唯一的编码方式来对级联的字符串序列进行编码。例如,使用 ISO/IEC 8825-1 中参考文献[1]定义的编码规则。

5 一般模型和要求

本章定义了采用群组公钥的匿名数字签名机制的一般模型和要求。此外,本章还规定了与采用群组公钥的匿名签名机制相关的特定要求。

本部分凡涉及密码算法相关的内容,按国家有关法规实施;凡涉及采用密码技术解决机密性、完整性、真实性、抗抵赖需求的应遵循密码相关国家标准和行业标准。

本部分使用群组公钥签名的匿名机制包含了群组和群组成员。每个群组应包含相关的群组成员发布方。根据具体的机制,可能还拥有群组成员打开方和/或群组签名连接方。多个实体可能承担群组成员打开方或群组签名连接方的作用。机制的匿名程度依赖于匿名强度(例如群组的大小)、是否拥有打开能力、是否拥有连接能力、怎样进行撤销、是否发布方知道私钥和泄露私钥的可能性。本部分规定的机制所使用的对象标识符见附录 A,所对应的数值实例参见附录 E。

基于群组的匿名数字签名机制包括下列规定的过程:

- a) 密钥生成过程;
- b) 签名过程;
- c) 验证过程;
- d) 打开过程(如果机制支持打开能力);
- e) 连接过程(如果机制支持连接能力);
- f) 撤销过程。

本部分用到的撤销方法的对比分析参见附录 D。

本部分规定的采用群组公钥的匿名数字签名机制包含了各种实体。一些存在于所有机制中,另一些只存在于部分机制中。这些实体如下:

- 签名方:签名方是产生签名的实体。在一些机制中,签名方分割为两个实体。例如,在直接匿名机制中,签名方的角色分割为带有限计算和存储能力的主签名方,例如可信平台模块(TPM)和带高计算能力但低安全容错性的辅助签名方,如普通的计算平台。
- 验证方:验证方是验证数字签名的实体。
- 群组成员发布方:群组成员发布方是发布群组成员证书给签名方的实体。该实体存在于本部分所有的机制中。
- 群组成员打开方:群组成员打开方是有能力从签名中确认签名方身份的实体。该实体存在于本部分规定的某些机制中。
- 群组成员连接方:群组成员连接方能够验证两个签名是否由同一个签名方使用连接密钥或连接基产生的。该实体存在于本部分规定的某些机制中。

使用本部分中的任意机制,均应满足下列条件:

- 包含在匿名数字签名机制的每个实体都知道一组共同的群组参数,它被用来计算机制中的不同函数;
- 每个验证方可以访问群组验证密钥的真实副本;
- 在发布群组签名密钥时,签名方和群组成员发布方需要有可靠地通道来确保群组成员发布方

- 只能提供群组成员签名密钥给合法的群组成员；
- 抗碰撞密码杂凑函数应采用 GB/T 32905 中规定的函数；
- 强健的椭圆曲线生成器，如 ISO/IEC 15946-5 中规定的一种，应在一些匿名数字签名机制中使用；
- 强健的随机数生成器，如 GB/T 32905 中规定的一种，应在一些匿名数字签名机制中使用。

6 具有连接能力的机制

6.1 概述

本章规定了具有连接能力的四种数字签名机制。

注 1：在参考文献中 6.2 的机制称为单签名方案，同时，6.3、6.4 和 6.5 的方案称为直接匿名证明(DAA)方案。6.2、6.4 和 6.5 中的机制和相关安全证明分别基于参考文献[9]、[6]和[11]。6.3 的机制基于参考文献[3]中的方案，它是对参考文献[4]方案的小的修改，并且在参考文献[4]全文中给出其相应的安全分析。

注 2：对于验证等特定的应用，本章所规定的匿名签名机制，在进行签名之前，被签名消息可能被进行密码杂凑运算或/和被连接额外的信息。

6.2 机制 1

6.2.1 符号

下列符号适用于本机制。

- $l_p, k, l_x, l_e, l_E, l_X, \epsilon$: 安全参数；
- p', q', e : 素数；
- $a, a_0, g, h, b, C_1, D, C_2, d', d_1, d_2, t', t_1, t_2, A, f, T_1, T_2, T_3, T_4, d_3, d_4, d_5, t_3, t_4, t_5$: QR(n)内的整数；
- x', α, β : $[0, 2^{l_x} - 1]$ 内的整数；
- w_1, w_2, w_3 : $[0, 2^{2l_p} - 1]$ 内的整数；
- $\hat{c}, \dot{c}, c', c, c'', c'''$: 长度为 k 比特的整数；
- $\tilde{r}$: 长度为 $(2l_p + 1)$ 比特的整数；
- $t_1, \hat{s}_1, r', r_1, r_2$: 长度为 $\lceil \epsilon \cdot (l_x + k) \rceil$ 比特的整数；
- $t_2, \hat{s}_2$: 长度为 $\lceil \epsilon \cdot (2l_p + k + 1) \rceil$ 比特的整数；
- x : 长度为 $(l_x + 1)$ 比特的整数；
- r_3 : 长度为 $\lceil \epsilon \cdot (l_x + 2l_p + k + 1) \rceil$ 的整数；
- s_0, s_1, s_2, s' : $[-2^{l_x + k}, 2^{\epsilon(l_x + k)} - 1]$ 内的整数；
- s_3 : $[-2^{l_x + 2l_p + k + 1}, 2^{\epsilon(l_x + 2l_p + k + 1)} - 1]$ 内的整数；
- r_4, r_5 : 长度为 $\lceil \epsilon \cdot (2l_p + k) \rceil$ 比特的整数；
- r_9, r_{10} : 长度为 $\lceil \epsilon \cdot (2l_p + l_e + k) \rceil$ 比特的整数；
- s_4, s_5 : $[-2^{2l_p + k}, 2^{\epsilon(2l_p + k)} - 1]$ 内的整数；
- s_9, s_{10} : $[-2^{2l_p + l_e + k}, 2^{\epsilon(2l_p + l_e + k)} - 1]$ 内的整数；
- H : 输出 k 比特消息摘要的密码杂凑函数；
- H_r : 输出 $(2l_p)$ 比特消息摘要的密码杂凑函数。

6.2.2 密钥产生过程

6.2.2.1 密钥产生过程由两部分组成：建立过程和群组成员发布过程。其中，发布过程是群组成员发布

方产生证明、群组公共参数、群组公钥和群组发布密钥的过程,群组成员发布过程是通过运行在群成员发布方和群成员之间的交互协议来产生唯一的群组成员签名。

6.2.2.2 建立过程由群组成员发布方通过以下步骤执行:

- a) 选择下列参数: $l_p, k, l_x, l_e, l_E, l_X, \epsilon$ 。
- b) 选择 RSA 模块满足 $n = pq$, 且 $p = 2p' + 1, q = 2q' + 1$, 这里 p, q, p', q' 都是素数并且 p' 和 q' 均为 l_p 比特。
- c) 选择二次剩余模 n 的群组的随机数 a 执行以下步骤:
 - 1) 在 Zn^* 内满足 $\gcd(g+1, n) = 1$ 和 $\gcd(g-1, n) = 1$;
 - 2) 计算 $a = g^2 \pmod{n}$ 。
- d) 在 $QR(n)$ 内选择区别于 a 的随机数 a_0 。
- e) 在 $QR(n)$ 内选择区别于 a 和 a_0 的随机数 g 。
- f) 在 $QR(n)$ 内选择区别于 a, a_0 和 g 的随机数 h 。
- g) 在 $QR(n)$ 内选择区别于 a, a_0, h 和 g 的随机数 b 。
- h) 群组成员发布方选择两个密码杂凑函数 $H: \{0, 1\}^* \rightarrow \{0, 1\}^k$ 和 $H_r: \{0, 1\}^* \rightarrow \{0, 1\}^{2l_p}$ 。附录 B 给出了如何去构建 H_r 的示例。
- i) 输出: 群组公共参数 $(l_p, k, l_x, l_e, l_E, l_X, \epsilon)$,
 群组公钥 (n, a, a_0, g, h, b) ,
 群组成员发布密钥 (p', q') 。

注: 推荐参数的示例参见附录 C 中的 C.2。

6.2.2.3 群组成员发布过程需要在主签名方和发布方之间建立一个安全的鉴别通信信道, 如何建立该信道不在本机制的范围之内规定, 群组成员发布过程包括以下步骤:

- a) 群组成员选取随机整数 $x' \in [0, 2^{l_x} - 1]$ 。
- b) 群组成员选取随机整数 $\tilde{r} \in [0, 2n - 1]$ 。
- c) 群组成员计算 $C_1 = g^{x'} h^{\tilde{r}} \pmod{n}$ 。
- d) 群组成员通过以下步骤产生一个在基 g 和 h 下 C_1 的 $(x', \tilde{r})$ 知识证明 U :
 - 1) 群组成员选取随机整数 $t_1 \in [0, 2^{\epsilon(l_x + k)} - 1]$;
 - 2) 群组成员选取随机整数 $t_2 \in [0, 2^{\epsilon(2l_p + k + 1)} - 1]$;
 - 3) 群组成员计算 $D = g^{t_1} h^{t_2}, D = g^{t_1} h^{t_2} \pmod{n}$;
 - 4) 群组成员计算 $\hat{c} = H(g \parallel h \parallel C_1 \parallel D)$;
 - 5) 群组成员计算 $\hat{s}_1 = t_1 - \hat{c} x'$;
 - 6) 群组成员计算 $\hat{s}_2 = t_2 - \hat{c} \tilde{r}$;
 - 7) $U = (\hat{c}, \hat{s}_1, \hat{s}_2)$ 。
- e) 群组成员发送 C_1 和 U 给群组成员发布方。
- f) 群组成员发布方从群组成员接收到 C_1 和 U 。
- g) 群组成员发布方验证 C_1 属于 $QR(n)$:
 群组成员发布方验证是否存在 $(C_1 | p) = 1$ 且 $(C_1 | q) = 1$ 。
 如果任意的一个验证失败, 群组成员发布方输出拒绝和禁止。
- h) 群组成员发布方验证知识证明 U :
 - 1) 群组成员发布方计算 $D' = g^{\hat{s}_1} h^{\hat{s}_2} C_1^{\hat{c}} \pmod{n}$;
 - 2) 群组成员发布方计算 $\hat{c} = H(g \parallel h \parallel C_1 \parallel D')$;
 - 3) 群组成员发布方验证 $\hat{c} = \hat{c}, \hat{s}_1 \in [-2^{l_x + k}, 2^{\epsilon(l_x + k)} - 1]$, 并且 $\hat{s}_2 \in [-2^{2l_p + k + 1}, 2^{\epsilon(2l_p + k + 1)} - 1]$ 。
 如果任意的一个验证失败, 群组成员发布方输出拒绝和禁止。

- i) 群组成员发布方选取随机的基数 $\alpha \in [0, 2^{l_x} - 1]$ 。
- j) 群组成员发布方选取随机整数 $\beta \in [0, 2^{l_x} - 1]$ 。
- k) 群组成员发布方发送 α 和 β 给群组成员。
- l) 群组成员从群组成员发布方接收到 α 和 β 。
- m) 群组成员计算 $x = 2^{l_x} + [\alpha x' + \beta \pmod{2^{l_x}}]$ 。
- n) 群组成员计算 $C_2 = a^x \pmod{n}$ 。
- o) 群组成员计算 $v = (\alpha x' + \beta) | 2^{l_x}$ 。
- p) 群组成员产生在基 a 下 C_2 的离散对数 x 的知识证明 V :
 - 1) 群组成员发布方选取随机整数 $r' \in [0, 2^{\epsilon(l_x+k)} - 1]$;
 - 2) 群组成员计算 $d' = a^{r'} \pmod{n}$;
 - 3) 群组成员计算 $c' = H(a \| g \| C_2 \| d')$;
 - 4) 群组成员计算 $s' = r' - c'(x - 2^{l_x})$;
 - 5) 群组成员使得 $V = (c', s')$ 。
- q) 群组成员产生知识证明 W :
 - 1) 群组成员选取随机整数 $r_1 \in [0, 2^{\epsilon(l_x+k)} - 1]$;
 - 2) 群组成员选取随机整数 $r_2 \in [0, 2^{\epsilon(l_x+k)} - 1]$;
 - 3) 群组成员选取随机整数 $r_3 \in [0, 2^{\epsilon(l_x+2l_p+k+1)} - 1]$;
 - 4) 群组成员计算 $d_1 = a^{r_1} \pmod{n}$;
 - 5) 群组成员计算 $d_2 = g^{r_1} (g^l)^{r_2} h^{r_3} \pmod{n}$, 其中 $l = 2^{l_x}$;
 - 6) 群组成员计算 $c = H(a \| g \| h \| C_1 \| C_2 \| d_1 \| d_2)$;
 - 7) 群组成员计算 $s_1 = r_1 - c(x - 2^{l_x})$;
 - 8) 群组成员计算 $s_2 = r_2 - cv$;
 - 9) 群组成员计算 $s_3 = r_3 - ca\tilde{r}$;
 - 10) 群组成员计算 $W = (c, s_1, s_2, s_3)$ 。
- r) 群组成员发送 C_2 、 V 和 W 给群组成员发布方。
- s) 群组成员发布方从群组成员接收到 C_2 、 V 和 W 。
- t) 群组成员发布方检查 C_2 是否属于 $QR(n)$ 。
群组成员发布方检查是否满足 $(C_2 | p) = 1$ 且 $(C_2 | q) = 1$ 。如果任意一个验证失败, 群组成员发布方输出拒绝和禁止。
- u) 群组成员发布方验证知识证明 V :
 - 1) 群组成员发布方计算 $s_0 = s' - c' 2^{l_x}$;
 - 2) 群组成员发布方计算 $t' = C_2^{c'} a^{s_0} \pmod{n}$;
 - 3) 群组成员发布方计算 $c'' = H(a \| g \| C_2 \| t')$;
 - 4) 群组成员发布方检查 $c'' = c'$ 并且 $s' \in [-2^{l_x+k}, 2^{\epsilon(l_x+k)} - 1]$ 。如果任意一个验证失败, 群组成员发布方输出拒绝和禁止。
- v) 群组成员发布方验证知识证明 W :
 - 1) 群组成员发布方计算 $t_1 = (C_2 / a^L)^c a^{s_1} \pmod{n}$, 其中 $L = 2^{l_x}$;
 - 2) 群组成员发布方计算 $t_2 = (C_1^a g^\beta)^c g^{s_1} (g^l)^{s_2} h^{s_3} \pmod{n}$, 其中 $l = 2^{l_x}$;
 - 3) 群组成员发布方计算 $c''' = H(a \| g \| h \| C_1 \| C_2 \| t_1 \| t_2)$;
 - 4) 群组成员发布方验证 $c''' = c$, s_1 属于 $[-2^{l_x+k}, 2^{\epsilon(l_x+k)} - 1]$, s_2 属于 $[-2^{l_x+k}, 2^{\epsilon(l_x+k)} - 1]$ 并且 s_3 属于 $[-2^{l_x+2l_p+k+1}, 2^{\epsilon(l_x+2l_p+k+1)} - 1]$ 。如果任意一个验证失败, 群组成员发布方输出拒绝和禁止。
- w) 群组成员发布方选取随机数 $e \in [2^{l_E} - 2^{l_e} + 1, 2^{l_E} + 2^{l_e} - 1]$ 。

- x) 群组成员发布方计算 $d_1 = 1/e \pmod{p'q'}$ 。
- y) 群组成员发布方计算 $A = (a_0 C_2)^{d_1} \pmod{n}$ 。
- z) 群组成员发布方存储 $(A, e, Member)$ 到群组成员列表中。
- aa) 群组成员发布方发送 A 和 e 给群组成员。
- bb) 群组成员从群组成员发布方接收到 A 和 e 。
- cc) 群组成员验证 $A^e = a_0 a^x \pmod{n}$ 。
- dd) 签名方的群组成员签名密钥 (A, e, x) , 其中 x 是群组成员私钥, (A, e) 是群组成员凭证。

6.2.3 签名过程

连接基被用来实现连接能力,它由群组成员发布方或其他可信机构选取。输入群组公钥 (n, a, a_0, g, h, b) , 群组成员签名密钥 (A, e, x) , 连接基 bsn 和未签消息 $m \in \{0, 1\}^*$, 签名过程执行以下步骤:

- a) 群组成员计算 $f = (H_r(bsn))^2 \pmod{n}$;
- b) 群组成员选取随机整数 $w_1 \in [0, 2^{2l_p} - 1]$;
- c) 群组成员选取随机整数 $w_2 \in [0, 2^{2l_p} - 1]$;
- d) 群组成员选取随机整数 $w_3 \in [0, 2^{2l_p} - 1]$;
- e) 群组成员计算 $T_1 = Ab^{w_1} \pmod{n}$;
- f) 群组成员计算 $T_2 = g^{w_1} h^{w_2} \pmod{n}$;
- g) 群组成员计算 $T_3 = g^e h^{w_3} \pmod{n}$;
- h) 群组成员计算 $T_4 = f^x \pmod{n}$;
- i) 群组成员选取随机整数 $r_1 \in [0, 2^{\epsilon(l_e+k)} - 1]$;
- j) 群组成员选取随机整数 $r_2 \in [0, 2^{\epsilon(l_x+k)} - 1]$;
- k) 群组成员选取随机整数 $r_3 \in [0, 2^{\epsilon(2l_p+k)} - 1]$;
- l) 群组成员选取随机整数 $r_4 \in [0, 2^{\epsilon(2l_p+k)} - 1]$;
- m) 群组成员选取随机整数 $r_5 \in [0, 2^{\epsilon(2l_p+k)} - 1]$;
- n) 群组成员选取随机整数 $r_9 \in [0, 2^{\epsilon(2l_p+l_e+k)} - 1]$;
- o) 群组成员选取随机整数 $r_{10} \in [0, 2^{\epsilon(2l_p+l_e+k)} - 1]$;
- p) 群组成员计算 $d_1 = T_1^{r_1} / (a^{r_2} b^{r_9}) \pmod{n}$;
- q) 群组成员计算 $d_2 = T_2^{r_1} / (g^{r_9} h^{r_{10}}) \pmod{n}$;
- r) 群组成员计算 $d_3 = g^{r_3} h^{r_4} \pmod{n}$;
- s) 群组成员计算 $d_4 = g^{r_1} h^{r_5} \pmod{n}$;
- t) 群组成员计算 $d_5 = f^{r_2} \pmod{n}$;
- u) 群组成员计算 $c = H(a \parallel a_0 \parallel g \parallel h \parallel T_1 \parallel T_2 \parallel T_3 \parallel T_4 \parallel d_1 \parallel d_2 \parallel d_3 \parallel d_4 \parallel d_5 \parallel m)$;
- v) 群组成员计算 $s_1 = r_1 - c(e - 2^{l_e})$;
- w) 群组成员计算 $s_2 = r_2 - c(x - 2^{l_x})$;
- x) 群组成员计算 $s_3 = r_3 - cw_1$;
- y) 群组成员计算 $s_4 = r_4 - cw_2$;
- z) 群组成员计算 $s_5 = r_5 - cw_3$;
- aa) 群组成员计算 $s_9 = r_9 - cew_1$;
- bb) 群组成员计算 $s_{10} = r_{10} - cew_2$;
- cc) 群组成员设置签名为 $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ 。

6.2.4 验证过程

输入消息 m 、连接基 bsn 、签名 $(c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ 以及群组公钥 (n, a, a_0, g, h, b) ,

g, h, b), 验证过程执行下列步骤:

- 验证方计算 $f = (H_r(bsn))^2 \pmod n$;
- 验证方计算 $t_1 = a_0^c T_1^{s_1 - cl'} / (a^{s_2 - cl} b^{s_9}) \pmod n$, 其中 $l' = 2^{l_E}, L = 2^{l_x}$;
- 验证方计算 $t_2 = T_2^{s_1 - cl'} / (g^{s_9} h^{s_{10}}) \pmod n$, 其中 $l' = 2^{l_E}$;
- 验证方计算 $t_3 = T_2^c g^{s_3} h^{s_4} \pmod n$;
- 验证方计算 $t_4 = T_3^c g^{s_1 - cl'} h^{s_5} \pmod n$, 其中 $l' = 2^{l_E}$;
- 验证方计算 $t_5 = T_4^c f^{s_2 - cl} \pmod n$, 其中 $L = 2^{l_x}$;
- 验证方计算 $c' = H(a \parallel a_0 \parallel g \parallel h \parallel T_1 \parallel T_2 \parallel T_3 \parallel T_4 \parallel t_1 \parallel t_2 \parallel t_3 \parallel t_4 \parallel t_5 \parallel m)$;
- 如果 $c' = c$, s_1 在 $[-2^{l_e + k}, 2^{\epsilon(l_e + k)} - 1]$, s_2 在 $[-2^{l_x + k}, 2^{\epsilon(l_x + k)} - 1]$, s_3 在 $[-2^{2l_p + k}, 2^{\epsilon(2l_p + k)} - 1]$, s_4 在 $[-2^{2l_p + k}, 2^{\epsilon(2l_p + k)} - 1]$, s_5 在 $[-2^{2l_p + k}, 2^{\epsilon(2l_p + k)} - 1]$, s_9 在 $[-2^{2l_p + l_e + k}, 2^{\epsilon(2l_p + l_e + k)} - 1]$, s_{10} 在 $[-2^{2l_p + l_e + k}, 2^{\epsilon(2l_p + l_e + k)} - 1]$, 那么返回 1 (正确);
- 否则返回 0 (不正确)。

6.2.5 连接过程

有两个使用同一个连接基 bsn 计算的合法的签名 $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ 和 $\sigma' = (c', s_1', s_2', s_3', s_4', s_5', s_9', s_{10}', T_1', T_2', T_3', T_4')$, 该连接过程执行下列步骤:

如果 $T_4 = T_4'$, 输出 1 (已连接), 否则输出 0 (未连接)。

6.2.6 撤销过程

该机制撤销过程的细节参见参考文献[10]。在该机制中支持两个类型的撤销(私钥撤销和验证方黑名单撤销)。私钥撤销既可以是全局撤销也可以是本地撤销。验证方黑名单撤销属于本地撤销。

注 1: 撤销过程的细节参见参考文献[8]。

私钥撤销:

- 如果群组成员的签名密钥 (A, e, x) 被损坏, 群组成员发布方或者验证方把 x 放进该类型撤销列表 RL 中;
- 给定一个使用连接基 bsn 计算的合法的签名 $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ 和一个该类型的撤销列表 RL, 验证方可以按照如下方式检查该签名的撤销: 对于每一个 $x' \in \text{RL}$, 验证 $T_4 \neq (H_r(bsn))^{2x'} \pmod n$ 。如果验证失败, 输出 0 (已撤销); 否则输出 1 (有效)。

注 2: 只有群组成员发布方或验证方已经知道该群组成员签名密钥, 该私钥撤销才可以工作。

验证方黑名单撤销:

- 如果生成签名时使用连接基 bsn , 则验证方可以创建他自己的对应于 bsn 的撤销列表 RL。如果验证方想要将签名 $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ 的签名方放进黑名单, 需要把 T_4 放入该类型的撤销名单 RL 中。
- 给定一个使用连接基 bsn 计算的合法的签名 $\sigma = (c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$ 和该类型的撤销列表 RL, 验证方可以检查该签名的撤销如下: 对于每一个 $T_4' \in \text{RL}$, 验证 $T_4 \neq T_4'$ 。如果验证失败, 输出 0 (已撤销); 否则输出 1 (有效)。

注 3: 为了在本机制中使用验证者黑名单撤销方法, 签名者对于每个验证者都使用特定的连接基。连接基可由验证者选择, 或者由签名者和验证者提前商定。

6.3 机制 2

6.3.1 符号

下列符号适用于本机制。

—— $l_n, l_f, l_e, l'_e, l_v, l_\phi, l_H, l_r, l_s, l_\Gamma, l_\rho$: 安全参数;

- p', q', ρ, Γ, e : 素数;
- $g', g, h, S, Z, R_0, R_1, U, U', A, A', T, T_t'$: $QR(n)$ 内的整数;
- $x_0, x_1, x_z, x_s, x_h, x_g, s_e, r_e$: $[1, p' \cdot q']$ 内的整数;
- $\gamma, J_l, K_l, K_l', J, K, J', K'$: 其乘法阶模 Γ 的整数为 ρ ;
- f, f' : $[0, \rho - 1]$ 内的整数;
- f_0, f_1 : 长度为 l_f 比特的整数;
- c_h, c, c', n_l, n_v : 长度为 l_H 比特的整数;
- n_T, n_H : 长度为 l_ϕ 比特的整数;
- t, t_2, r_f : 长度为 $(2l_f + l_\phi + l_H + 1)$ 比特的整数;
- t_1 : 长度为 $(l_e + l_H)$ 比特的整数;
- r_0, r_1 : 长度为 $(l_f + l_\phi + l_H)$ 比特的整数;
- r_v : 长度为 $(l_v + l_\phi + l_H)$ 比特的整数;
- r_{v*} : 长度为 $(l_e + l_n + 2l_\phi + l_H + 1)$ 比特的整数;
- $r_{v'}$: 长度为 $(l_n + 2l_\phi + l_H)$ 比特的整数;
- s_v' : 长度为 $(l_n + 2l_\phi + l_H + 1)$ 比特的整数;
- s_0, s_1 : 长度为 $(l_f + l_\phi + l_H + 1)$ 比特的整数;
- v', w : 长度为 $(l_n + l_\phi)$ 比特的整数;
- v^* : 长度为 $(l_v - 1)$ 比特的整数;
- v'' : 长度为 l_v 比特的整数;
- v : 长度为 $(l_n + l_\phi + l_v + 1)$ 比特的整数;
- bsn_l : 群组成员发布方的连接基;
- H_r : 输出长度为 $(l_r + l_\phi)$ 比特的消息摘要的密码杂凑函数。

6.3.2 密钥生成过程

6.3.2.1 密钥产生过程由两部分组成: 建立过程和群组成员发布过程。其中, 发布过程是群组成员发布方产生群组公共参数、群组公钥和群组发布密钥的过程, 群组成员发布过程是通过运行在群成员发布方和群成员之间的交互协议来产生唯一的群组成员签名。在群组成员发布过程的撤销验证要求能够防止重新放置撤销的群组成员私钥。

6.3.2.2 该建立过程由群组成员发布方执行下列步骤:

- a) 选取下列参数: $l_n, l_f, l_e, l'_e, l_v, l_\phi, l_H, l_r, l_s, l_\Gamma, l_\rho$;
- b) 选取两个密码杂凑函数 $H: \{0, 1\}^* \rightarrow \{0, 1\}^{l_H}$ 和 $H_r: \{0, 1\}^* \rightarrow \{0, 1\}^{l_r + l_\phi}$; 如何构建 H_r 的示例在附录 B 给出;
- c) 选取 RSA 模块 $n = pq$ 使得 $p = 2p' + 1, q = 2q' + 1$, 其中 p, q, p', q' 是素数, n 的长度为 l_n 比特;
- d) 在 $QR(n)$ 中选取随机整数 g' ;
- e) 在 $[1, p' \cdot q']$ 中选取随机整数 $x_0, x_1, x_z, x_s, x_h, x_g$;
- f) 计算 $g = (g')^{x_g} \bmod n, h = (g')^{x_h} \bmod n, S = h^{x_s} \bmod n$;
- g) 计算 $Z = h^{x_z} \bmod n, R_0 = S^{x_0} \bmod n, R_1 = S^{x_1} \bmod n$;
- h) 选取长度为 l_ρ 比特的素数 ρ ;
- i) 选取长度为 l_Γ 比特的素数 Γ , 使得 $\Gamma - 1$ 是 ρ 的倍数并且 $(\Gamma - 1)/\rho$ 不是 ρ 的倍数;
- j) 选取随机数 γ , 其乘法阶模 Γ 的整数为 ρ ;
- k) 输出如下: 群组公共参数 $= (l_n, l_f, l_e, l'_e, l_v, l_\phi, l_H, l_r, l_s, l_\Gamma, l_\rho, H, H_r)$,
群组公钥 $= (n, g', g, h, S, Z, R_0, R_1, \gamma, \Gamma, \rho)$,

群组成员发布密钥 $= (p', q')$ 。

注：提供一个推荐的参数示例参见附录 C 中的 C.2。

6.3.2.3 群组成员发布过程需要在主签名方和发布方之间建立一个安全的鉴别通信信道，如何建立该信道不在本机制的范围之内规定，群组成员发布过程包括以下步骤：

- a) 主签名方和群组成员发布方之间协商一个连接基 bsn_I ；
- b) 辅助签名方计算 $J_I = (H_r(1 \parallel bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$ 并发送 J_I 给主签名方；

注 1：如参考文献[19]中所述，保持签名方匿名的前提是在任何的 DAA 签名中 bsn_I 都不会作为 bsn 使用。在无法保证这一前提的应用中，将 $J_I = (H_r(1 \parallel bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$ 改为 $J_I = (H_r(0 \parallel bsn_I))^{(\Gamma-1)/\rho} \bmod \Gamma$ 可避免该潜在的问题。

- c) 主签名方验证 $(J_I)^\rho \equiv 1 \pmod{\Gamma}$ ；
- d) 主签名方选择随机数 $f \in [0, \rho-1]$ 或从它的密钥种子值导出 f ；
- e) 主签名方计算 $f_1 = f/2^{l_f}$ ；
- f) 主签名方计算 $f_0 = f - 2^{l_f} \cdot f_1$ ，该 (f_0, f_1) 对是群组成员私钥的一部分；
- g) 主签名方随机选取一个长度为 $(l_n + l_\phi)$ 比特的整数 v' ；
- h) 主签名方计算 $U = (R_0^{f_0} \cdot R_1^{f_1} \cdot S^{v'}) \bmod n$ ；
- i) 主签名方计算 $K_I = (J_I)^f \bmod \Gamma$ ；
- j) 主签名方发送 U 和 K_I 给辅助签名方，然后辅助签名方能够将它们发送给群组成员发布方；

注 2：在以上步骤后，群组成员发布方对每一个撤销名单上的 (f_0, f_1) 可计算 $f = f_0 + f_1 \cdot 2^{l_f}$ 并验证 $KI \neq J_I^f \bmod \Gamma$ 。如果签名方被撤销，群组成员发布方禁止群组成员发布过程。

- k) 主签名方随机选取两个长度为 $(l_f + l_\phi + l_H)$ 比特的整数 r_0, r_1 ；
- l) 主签名方随机选取一个长度为 $(l_n + 2l_\phi + l_H)$ 比特的整数 r_v' ；
- m) 主签名方计算 $U' = (R_0^{r_0} \cdot R_1^{r_1} \cdot S^{r_v'}) \bmod n$ ；
- n) 主签名方计算 $r_f = r_0 + r_1 \cdot 2^{l_f}$ ；
- o) 主签名方计算 $K_I' = (J_I)^{r_f} \bmod \Gamma$ ；
- p) 签名方发送 U' 和 K_I' 给辅助签名方；
- q) 群组成员发布方选取随机数 $n_I \in \{0, 1\}^{l_H}$ 并发送 n_I 给辅助签名方；
- r) 辅助签名方计算 $c_h = H(n \parallel R_0 \parallel R_1 \parallel S \parallel U \parallel K_I \parallel U' \parallel K_I' \parallel n_I)$ ；
- s) 辅助签名方发送 c_h 给主签名方；
- t) 主签名方选取随机数 $n_T \in \{0, 1\}^{l_\phi}$ ；
- u) 主签名方计算 $c = H(c_h \parallel n_T)$ ；
- v) 主签名方计算 $s_v' = r_v' + c \cdot v'$ ， $s_0 = r_0 + c \cdot f_0$ ， $s_1 = r_1 + c \cdot f_1$ ；
- w) 主签名方发送 (c, n_T, s_0, s_1, s_v') 给辅助签名方；
- x) 辅助签名方发送 (c, n_T, s_0, s_1, s_v') 给群组成员发布方；
- y) 群组成员发布方验证 s_0 和 s_1 至多是长度为 $(l_f + l_\phi + l_H + 1)$ 比特的整数；
- z) 群组成员发布方验证 s_v' 至多是长度为 $(l_n + 2l_\phi + l_H + 1)$ 比特整数；
- aa) 群组成员发布方计算 $U' = (U^{-c} \cdot R_0^{s_0} \cdot R_1^{s_1} \cdot S^{s_v'}) \bmod n$ ；
- bb) 群组成员发布方计算 $t = s_0 + s_1 \cdot 2^{l_f}$ ；
- cc) 群组成员发布方计算 $K_I' = (K_I^{-c} \cdot J_I^t) \bmod \Gamma$ ；
- dd) 群组成员发布方验证 $c = H(H(n \parallel R_0 \parallel R_1 \parallel S \parallel U \parallel K_I \parallel U' \parallel K_I' \parallel n_I) \parallel n_T)$ ；
- ee) 群组成员发布方随机选取长度为 $(l_v - 1)$ 比特的 v^* ；
- ff) 群组成员发布方随机从 $[2^{l_e-1}, 2^{l_e-1} + 2^{l_e-1}]$ 中选取一个素数 e ；
- gg) 群组成员发布方计算 $v'' = v^* + 2^{l_v-1}$ ；
- hh) 群组成员发布方计算 $A = (Z \cdot U^{-1} \cdot S^{-v''})^{1/e} \bmod n$ ；

- ii) 辅助签名方选取随机数 $n_H \in \{0,1\}^{l_\phi}$ 并发送给群组成员发布方;
- jj) 群组成员发布方从 $[0, p' \cdot q']$ 中随机选取 r_e ;
- kk) 群组成员发布方计算 $A' = (Z \cdot U^{-1} \cdot S^{-v''})^{r_e} \bmod n$;
- ll) 群组成员发布方计算 $c' = H(n \parallel Z \parallel S \parallel U \parallel v'' \parallel A \parallel A' \parallel n_H)$;
- mm) 群组成员发布方计算 $s_e = (r_e - c'/e) \bmod p' \cdot q'$;
- nn) 群组成员发布方发送 c', s_e 和 (A, e, v'') 给辅助签名方;
- oo) 辅助签名方验证 e 是 $[2^{l_e-1}, 2^{l_e-1} + 2^{l'_e-1}]$ 内的一个素数;
- pp) 辅助签名方计算 $A' = (A^{c'} \cdot (Z \cdot U^{-1} \cdot S^{-v''})^{s_e}) \bmod n$;
- qq) 辅助签名方验证 $c' = H(n \parallel Z \parallel S \parallel U \parallel v'' \parallel A \parallel A' \parallel n_H)$;
- rr) 辅助签名方发送 v'' 给主签名方;
- ss) 主签名方计算 $v = v' + v''$ 并存储 (f_0, f_1, v) , 同时辅助签名方存储 (A, e) ;
- tt) 签名方的群组成员签名密钥为 (f_0, f_1, A, e, v) , 其中, (f_0, f_1, v) 是群组成员私钥, (A, e) 是群组成员凭证。

6.3.3 签名过程

输入群组公钥 $(n, g', g, h, S, Z, R_0, R_1, \gamma, \Gamma, \rho)$ 、群组成员签名密钥 (f_0, f_1, A, e, v) 、连接基 bsn 、随机数 $n_V \in \{0,1\}^{l_H}$ 和未签消息 $m \in \{0,1\}^*$, 其中, 连接基既可以是特殊的字符 $\perp$, 也可以是任意的字符串并由其来实现连接能力, 它由群组成员发布方或验证方选取, 或者由两者之前协商选取。签名过程执行以下步骤:

注 1: 随机数 n_V 通常由验证方选取;

注 2: 另外一种处理 n_V 的方法是将 n_V 作为消息 m 的一部分。

- a) 主签名方拥有群组签名私钥 (f_0, f_1, v) , 同时辅助的签名方拥有 (A, e) ;
- b) 如果 $bsn = \perp$, 辅助的签名方从 $[0, \rho-1]$ 选取随机数 t 并计算 $J = (\gamma)^t \bmod \Gamma$;
- c) 如果 $bsn \neq \perp$, 辅助的签名方计算 $J = (H_\Gamma(1 \parallel bsn))^{(\Gamma-1)/\rho} \bmod \Gamma$;
- d) 辅助的签名方发送 J 给主签名方;
- e) 主签名方验证 $(J)^\rho \equiv 1 \bmod \Gamma$;
- f) 主签名方计算 $f = f_0 + f_1 \cdot 2^{l_f}$;
- g) 主签名方计算 $K = (J)^f \bmod \Gamma$;
- h) 主签名方随机选取一个长度为 $(l_v + l_\phi + l_H)$ 比特的整数 r_v ;
- i) 主签名方随机选取两个长度为 $(l_f + l_\phi + l_H)$ 比特的整数 r_0, r_1 ;
- j) 主签名方计算 $T_t' = (R_0^{r_0} \cdot R_1^{r_1} \cdot S^{r_v}) \bmod n$;
- k) 主签名方计算 $r_{f'} = (r_0 + r_1 \cdot 2^{l_f}) \bmod \rho$;
- l) 主签名方计算 $K' = (J)^{r_{f'}} \bmod \Gamma$;
- m) 主签名方发送 K, T_t', K' 给辅助签名方;
- n) 辅助签名方随机选取长度为 $(l_n + l_\phi)$ 比特的整数 w ;
- o) 辅助签名方随机选取长度为 $(l'_e + l_\phi + l_H)$ 比特的整数 r_e ;
- p) 辅助签名方随机选取长度为 $(l_e + l_n + 2l_\phi + l_H + 1)$ 比特的整数 r_{v^*} ;
- q) 辅助签名方计算 $T = (A \cdot S_w) \bmod n$;
- r) 辅助签名方计算 $T' = (T_t' \cdot T^{r_e} \cdot S^{r_{v'}}) \bmod n$;
- s) 辅助签名方计算 $c_h = H(n \parallel R_0 \parallel R_1 \parallel S \parallel Z \parallel \gamma \parallel \Gamma \parallel \rho \parallel J \parallel T \parallel K \parallel T' \parallel K' \parallel n_V)$;
- t) 辅助签名方 c_h 给主签名方;
- u) 主签名方选择随机数 $n_T \in \{0,1\}^{l_\phi}$;
- v) 主签名方计算 $c = H(H(c_h \parallel n_T) \parallel m)$;

- w) 主签名方计算 $s_v = r_v + c \cdot v, s_0 = r_0 + c \cdot f_0, s_1 = r_1 + c \cdot f_1$;
- x) 主签名方发送 (c, n_T, s_v, s_0, s_1) 给辅助签名方;
- y) 主签名方计算 $s_e = r_e + c \cdot (e - 2^{l_e - 1})$;
- z) 主签名方计算 $s_v' = s_v + r_{v*} - c \cdot w \cdot e$;
- aa) 辅助签名方输出群组签名 $\sigma = (J, K, T, c, n_T, s_v', s_0, s_1, s_e)$ 。

6.3.4 验证过程

输入消息 m 、连接基 bsn 、随机数 $n_V \in \{0, 1\}^{l_H}$ 、签名 $(J, K, T, c, n_T, s_v', s_0, s_1, s_e)$ 以及群组公钥 $(n, g', g, h, S, Z, R_0, R_1, \gamma, \Gamma, \rho)$, 验证过程执行下列步骤:

- a) 验证 $(J)^\rho \equiv 1 \pmod{\Gamma}$ 和 $(K)^\rho \equiv 1 \pmod{\Gamma}$;
- b) 验证 s_0 和 s_1 是长度不超过 $(l_f + l_\phi + l_H + 1)$ 比特的整数;
- c) 验证 s_e 是长度不超过 $(l_e' + l_\phi + l_H + 1)$ 比特的整数;
- d) 计算 $t_1 = s_e + c \cdot 2^{l_e - 1}$;
- e) 计算 $t_2 = s_0 + s_1 \cdot 2^{l_f}$;
- f) 计算 $T' = (Z^{-c} \cdot T^{t_1} \cdot R_0^{s_0} \cdot R_1^{s_1} \cdot S^{s_v'}) \pmod{n}$;
- g) 计算 $K' = (K^{-c} \cdot J^{t_2}) \pmod{\Gamma}$;
- h) 验证 $c = H(H(H(n \parallel R_0 \parallel R_1 \parallel S \parallel Z \parallel \gamma \parallel \Gamma \parallel \rho \parallel J \parallel T \parallel K \parallel T' \parallel K' \parallel n_V) \parallel n_T) \parallel m)$;
- i) 如果 J 是由连接基 bsn 导出的, 验证 $J = (H_\Gamma(1 \parallel bsn))^{(\Gamma-1)/\rho} \pmod{\Gamma}$;
- j) 撤销检查过程是可选的;
- k) 如果以上任意的一个验证失败, 输出 0(无效), 否则输出 1(有效)。

注: 验证过程中的验证检查可以在第一步骤中, 而不是该过程的最后步骤来进行。

6.3.5 连接过程

给定两个签名 $\sigma = (J, K, T, c, n_T, s_v', s_0, s_1, s_e)$ 和 $\sigma' = (J', K', T', c', n_T', s_v', s_0', s_1', s_e')$, 该连接过程执行下列步骤:

如果 $J = J'$ 和 $K = K'$, 输出 1(已连接), 否则输出 0(未连接)。

注: 如果连接过程因 $J \neq J'$ 输出 0, 意味着连接过程不能确定是否两个签名由同一个群组成员创建。

6.3.6 撤销过程

该机制撤销过程的细节参见参考文献[10]。在该机制中支持两个类型的撤销(私钥撤销和验证方黑名单撤销)。私钥撤销既可以是全局撤销也可以是本地撤销。验证方黑名单撤销属于本地撤销。

私钥撤销:

- 如果群组成员的签名密钥 (f_0, f_1, A, e, v) 被损坏, 群组成员发布方或者验证方计算 $f = f_0 + f_1 \cdot 2^{l_f}$ 并把 f 放进该类型撤销列表 RL 中;
- 给定签名 $\sigma = (J, K, T, c, n_T, s_v', s_0, s_1, s_e)$ 和一个该类型的撤销列表, 验证方可以检查该签名的撤销如下: 对于每一个 $f' \in \text{RL}$, 验证 $K \neq (J)^{f'} \pmod{\Gamma}$ 。如果验证失败, 输出 0(已撤销), 否则输出 1(有效)。

注 1: 只有群组成员发布方或验证方已经知道该群组成员损害的群组成员签名密钥, 该私钥撤销才可以工作。

验证方黑名单撤销:

- 如果生成签名时使用连接基 bsn , 并且验证方可以创建他自己的对应于 bsn 的撤销列表 RL。如果一个验证方想要将签名 $\sigma = (J, K, T, c, n_T, s_v', s_0, s_1, s_e)$ 的签名方放进黑名单, 则需要把 K 放入该类型的撤销名单 RL 中。

——给定签名 $\sigma = (J, K, T, c, n_T, s_v', s_0, s_1, s_e)$ 和一个该类型的撤销列表 RL, 验证方可以检查该签名的撤销如下: 对于每一个 $f' \in \text{RL}$, 验证 $K \neq K'$ 。如果任意的验证失败, 输出 0 (已撤销); 否则输出 1 (有效)。

注 2: 为了在本机制中使用验证者黑名单撤销方法, 签名者对于每个验证者都使用特定的连接基。连接基可由验证者选择, 或者有签名者和验证者提前商定。

6.4 机制 3

6.4.1 符号

下列符号适用于本机制。

- t : 安全参数;
- $Q_1, Q_2, A, F, R, J, K, J', K', T, R_1, R_2, R_3$: G_1 中的元素;
- W : G_2 中的元素;
- T_1, T_2, T_3, T_4, R_2 : G_T 中的元素;
- $y, f, f', x, r, c, c_h, a, b, r_f, r_x, r_a, r_b, s_f, s_x, s_a, s_b, u, v, r_u, r_v, s_u, s_v$: Z_p 中的整数;
- n_I, n_V, n_T : 长度为 t 比特的整数;
- H_1 : 输出元素在 Z_p 内的密码杂凑函数;
- H_2 : 输出元素在 G_1 内的密码杂凑函数。

6.4.2 密钥生成过程

6.4.2.1 密钥生成过程由群组成员发布方执行下列步骤:

- a) 选取一个大素数阶为 p 的非对称的双线性群组对 (G_1, G_2) 并且满足一个双线性函数 $e: G_1 \times G_2 \rightarrow G_T$ 。
- b) 选取 G_1 的生成元 P_1 。
- c) 选取 G_2 的生成元 P_2 。
- d) 选取两个密码杂凑函数 $H_1: \{0, 1\}^* \rightarrow Z_p$ 和 $H_2: \{0, 1\}^* \rightarrow G_1$ 。如何去构建该密码杂凑函数的示例在附录 B 中给出。
- e) 选取 G_1 中的随机元素 Q_1, Q_2 。
- f) 在 Z_p^* 内选取随机整数 y 并计算 $W = [y]P_2$ 。
- g) 计算 $T_1 = e(P_1, P_2)$, $T_2 = e(Q_1, P_2)$, $T_3 = e(Q_2, P_2)$ 以及 $T_4 = e(Q_2, W)$ 。
- h) 输出:
 群组公共参数 $= (G_1, G_2, G_T, p, e, P_1, P_2, H_1, H_2)$,
 群组公钥 $= (Q_1, Q_2, W, T_1, T_2, T_3, T_4)$,
 群组成员发布密钥 $= y$ 。

注 1: T_1, T_2, T_3 和 T_4 在群组公钥内是可选的, 他们可以从 P_1, P_2, Q_1, Q_2 中计算得出, 并且 W 由签名方和验证方计算。

注 2: 推荐参数的示例参见附录 C 中的 C.2。

6.4.2.2 群组成员发布过程需要在主签名方和发布方之间建立一个安全的鉴别通信信道, 如何建立该信道不在本机制的范围之内规定, 对于每一个签名方, 群组成员发布方产生群组成员签名如下:

- a) 从 Z_p^* 中选取两个随机数 f, x 或者从密钥种子值导出;
- b) 计算 $A = [1/(x+y)](P_1 + [f]Q_1)$;
- c) 输出群组成员签名密钥 (f, A, x) 。

注: 如果群组成员签名密钥使用以上的方法生成, 该机制提供对应群组成员发布方的连接能力, 除非群组成员发布方删除所有之前配置给签名方的群组成员签名密钥。

另外,通过群组成员发布方去获取一个群组成员签名密钥,每个签名方都可以运行群组成员发布过程:

- a) 群组成员发布方选取随机数 $n_I \in \{0,1\}^t$;
- b) 群组成员发布方发送 n_I 给主签名方;
- c) 签名方从 Z_p^* 中随机选取群组成员私钥 f 或从它的秘密种子值导出 f ;
- d) 签名方从 Z_p^* 中随机选取整数 r ;
- e) 签名方计算 $F = [f]Q_1$ 和 $R = [r]Q_1$;
- f) 签名方计算 $c = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel F \parallel R \parallel n_I)$;
- g) 签名方计算 $s = (r + c \cdot f) \bmod p$;
- h) 签名方发送 (F, c, s) 给群组成员发布方;
- i) 群组成员发布方计算 $R = [s]Q_1 - [c]F$;
- j) 群组成员发布方验证 $c = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel F \parallel R \parallel n_I)$;
- k) 群组成员发布方从 Z_p^* 中随机选取一个整数 x ;
- l) 群组成员发布方计算 $A = [1/(x+y)](P_1 + F)$;
- m) 群组成员发布方将 (A, x) 作为签名方的群组成员证书并发送给签名方;
- n) 签名方通过验证 $e(A, W + [x]P_2) = e(P_1 + F, P_2)$ 来验证该证书;
- o) 签名方的群组成员签名密钥为 (f, A, x) 。

6.4.3 签名过程

6.4.3.1 输入群组公钥 $(Q_1, Q_2, W, T_1, T_2, T_3, T_4)$ 、群组成员签名密钥 (f, A, x) 、连接基 bsn 和未签消息 $m \in \{0,1\}^*$, 其中,连接基既可以是特殊的字符 $\perp$,也可以是任意的字符串并由其来实现连接能力。签名过程执行以下步骤:

- a) 如果 $bsn = \perp$, 从 G_1 中选取随机元素, 否则计算 $J = H_2(bsn)$;
 - b) 计算 $K = [f]J$;
 - c) 从 Z_p^* 中选取随机数 a 并计算 $b = (a \cdot x) \bmod p$;
 - d) 计算 $T = A + [a]Q_2$;
 - e) 从 Z_p^* 中随机选取四个整数 r_f, r_x, r_a, r_b ;
 - f) 计算 $R_1 = [r_f]J$;
 - g) 计算 $R_2 = e(A, P_2)^{-r_x} \cdot T_2^{r_f} \cdot T_3^{r_b - a \cdot r_x} \cdot T_4^{r_a}$;
- 注: $e(A, P_2)$ 可以有群组成员预计算并在每一个签名过程中重复使用。
- h) 计算 $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2) \parallel m)$;
 - i) 计算 $s_f = (r_f + c \cdot f) \bmod p, s_x = (r_x + c \cdot x) \bmod p$;
 - j) 计算 $s_a = (r_a + c \cdot a) \bmod p, s_b = (r_b + c \cdot b) \bmod p$;
 - k) 输出匿名数字签名 $\sigma = (J, K, T, c, s_f, s_x, s_a, s_b)$ 。

6.4.3.2 签名过程可由主签名方和辅助签名方共同执行如下步骤:

- a) 主签名方拥有群组成员私钥 f , 同时辅助签名方拥有 (A, x) ;
- b) 如果 $bsn = \perp$, 主签名方从 G_1 随机选取元素 J , 否则计算 $J = H_2(bsn)$;
- c) 主签名方计算 $K = [f]J$;
- d) 主签名方从 Z_p^* 中随机选取整数 r_f ;
- e) 主签名方计算 $R_1 = [r_f]J$ 和 $R_{2t} = [r_f]Q_1$;
- f) 主签名方发送 (J, K, R_1, R_{2t}) 给辅助签名方;
- g) 辅助签名方从 Z_p^* 中随机选取整数 a , 并且计算 $b = (a \cdot x) \bmod p$;
- h) 辅助签名方计算 $T = A + [a]Q_2$;

- i) 辅助签名方从 Z_p^* 中随机选取三个整数 r_x, r_a, r_b ;
- j) 辅助签名方计算 $R_2 = e(R_{2t} - [r_x]T + [r_b]Q_2, P_2) \cdot T_4^{r_a}$;
- k) 辅助签名方计算 $c_h = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2)$;
- l) 辅助签名方发送 c_h 给主签名方;
- m) 主签名方选取随机数 $n_T \in \{0, 1\}^t$;
- n) 主签名方计算 $c = H_1(c_h \parallel n_T \parallel m)$;
- o) 主签名方计算 $s_f = (r_f + c \cdot f) \bmod p$;
- p) 主签名方计算发送 (c, n_T, s_f) 给辅助的签名方;
- q) 辅助的签名方计算 $s_x = (r_x + c \cdot x) \bmod p, s_a = (r_a + c \cdot a) \bmod p, s_b = (r_b + c \cdot b) \bmod p$;
- r) 辅助的签名方输出匿名数字签名 $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ 。

注 1: 由主签名方选取的随机数 n_T 是可选的和可省略的。

注 2: 签名过程可能包含验证方的随机数 n_V , 其作为可选的输入。如果 n_V 包含在输入内, 签名的步骤 h) 过程计算 $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2 \parallel n_V) \parallel m)$ 和步骤 k) 加入签名过程计算 $c_h = H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2 \parallel n_V)$ 。

6.4.4 验证过程

输入消息 m 、连接基 bsn 、签名 $(J, K, T, c, s_f, s_x, s_a, s_b)$ 和群组公钥 $(Q_1, Q_2, W, T_1, T_2, T_3, T_4)$, 验证过程执行下列步骤:

- a) 验证 J, K, T 是 G_1 中的元素;
- b) 验证 s_f, s_x, s_a, s_b 是 Z_p 中的整数;
- c) 如果 $bsn \neq \perp$, 验证 $J = H_2(bsn)$;
- d) 计算 $R_1 = [s_f]J - [c]K$;
- e) 计算 $R_2 = e(T, [-s_x]P_2 - [c]W) \cdot T_1^c \cdot T_2^{s_f} \cdot T_3^{s_b} \cdot T_4^{s_a}$;
- f) 验证 $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2) \parallel m)$;
- g) 撤销检查过程(可选);
- h) 如果以上的任意一种验证失败, 输出 0(无效); 否则输出 1(有效)。

注: 如果随机数 n_T 包含在签名里, 验证过程步骤 f) 验证 $c = H_1(H_1(p \parallel P_1 \parallel P_2 \parallel Q_1 \parallel Q_2 \parallel W \parallel J \parallel K \parallel T \parallel R_1 \parallel R_2) \parallel n_T \parallel m)$ 。

6.4.5 连接过程

给定两个签名 $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ 和 $\sigma' = (J', K', T', c', n_T', s_f', s_x', s_a', s_b')$, 连接过程执行下列步骤:

如果 $J = J'$ 且 $K = K'$, 输出 1(连接), 否则输出 0(非连接)。

注: 如果连接过程因 $J \neq J'$ 输出 0, 则意味着连接过程不能确定是否两个签名由同一个群组成员创建。

6.4.6 撤销过程

该机制撤销过程的细节参见参考文献[10]。在该机制中支持三个类型的撤销(私钥撤销、验证方黑名单撤销和签名撤销)。私钥撤销和签名撤销既可以是一个全局撤销也可以是一个本地撤销。验证方黑名单撤销是一个本地撤销。

私钥撤销:

- 如果群组成员的签名密钥 (f, A, x) 被损坏, 群组成员发布方或者验证方把 f 放进该类型撤销列表 RL 中;
- 给定签名 $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ 和一个该类型的撤销列表, 验证方可以对于每一个

$f' \in \text{RL}$, 验证是否满足 $K \neq [f']J$ 。如果验证失败, 输出 0 (已撤销), 否则输出 1 (有效)。

注 1: 只有群组成员发布方或验证方已经知道该群组成员损害的群组成员签名密钥, 该私钥撤销才可以工作。

黑名单撤销:

- 如果生成签名时使用连接基 bsn , 并且验证方可以创建他自己的对应于 bsn 的撤销列表 RL。
- 如果一个验证方想要将签名 $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ 的签名方放进黑名单, 则需要把 K 放入该类型的撤销名单 RL 中。
- 给定签名 $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ 和该类型的撤销列表 RL, 验证方可以对于每一个 $K' \in \text{RL}$, 验证 $K \neq K'$ 。如果验证失败, 输出 0 (已撤销); 否则输出 1 (有效)。

签名撤销:

- 如果群组成员发布方或验证方确定签名 $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ 是由恶意的签名方创建, 并且没有匹配的群组签名密钥, 发布方或验证方把签名中的 (J, K) 对放入该类型的撤销列表 RL 中。
- 为了执行撤销检查, 签名方需要使用零知识证明去证明他之前没有创建 RL 内的任何 (J', K') 。更特别的是, 设 $\sigma = (J, K, T, c, n_T, s_f, s_x, s_a, s_b)$ 只是由签名方创建的签名, 对于列表 RL 中的每一个 (J', K') 对, 签名方在零知识的条件下证明 $[f]J = K$ 且 $[f]J' \neq K'$ 如下:
 - a) 签名方从 Z_p^* 中选取随机整数 u ;
 - b) 签名方计算 $v = (-f \cdot u) \bmod p$;
 - c) 签名方计算 $T = [u]K' + [v]J'$ 。如果 $T = O_E$, 证明已撤销;
 - d) 签名方从 Z_p^* 中选取随机整数 r_u, r_v ;
 - e) 签名方计算 $R_1 = [r_u]K + [r_v]J$ 和 $R_3 = [r_u]K' + [r_v]J'$;
 - f) 签名方计算 $c = H_1(p \parallel P_1 \parallel J \parallel K \parallel J' \parallel K' \parallel T \parallel R_1 \parallel R_3 \parallel m)$;
 - g) 签名方计算 $s_u = (r_u + c \cdot u) \bmod p$ 和 $s_v = (r_v + c \cdot v) \bmod p$;
 - h) 签名方计算发送 (T, c, s_u, s_v) 作为非撤销证明给验证方;
 - i) 验证方验证 $T \in G_1$ 并且 $s_u, s_v \in Z_p$;
 - j) 验证方验证 $T \neq O_E$;
 - k) 验证方计算 $R_1 = [s_u]K + [s_v]J$, $R_3 = [s_u]K' + [s_v]J' - [c]T$;
 - l) 验证方验证 $c = H_1(p \parallel P_1 \parallel J \parallel K \parallel J' \parallel K' \parallel T \parallel R_1 \parallel R_3 \parallel m)$;
 - m) 如果任意的验证步骤失败, 验证方拒绝非撤销证明, 否则, 接受证明。

注 2: 签名验证需要一个签名方和验证方之间的交互过程。

注 3: 为了保持匿名, 建议拥有一个可信实体来更新签名撤销列表。如果一个恶意的实体控制签名撤销列表, 签名方的匿名性就会降低。

6.5 机制 4

6.5.1 符号

下列符号适用于本机制。

- t : 安全参数;
- $Q_2, U', A, B, C, D, R, S, T, W, J, K, R_1, R_2, C': G_1$ 的元素;
- $X, Y, X': G_2$ 的元素;
- $x, y, f, u, v, w, v', l, c, r, h, s, x', \beta: Z_p$ 中的整数;
- n_1, n_v, n_T : 长度为 t 比特的整数;
- H_1 : 输出元素在 G_1 内的密码杂凑函数;
- H_2, H_3, H_4 : 输出元素在 Z_p 内的密码杂凑函数。

6.5.2 密钥生成过程

6.5.2.1 密钥生成过程包括两部分:建立过程和群组成员发布过程。建立过程由群组成员发布方执行,用于创建群公共参数、群组公钥以及群组成员发布密钥。群组成员发布过程是在群组成员发布方和群组成员之间运行的互操作协议,用于为群组成员创建唯一的群组成员签名密钥。

6.5.2.2 密钥生成过程由群组成员发布方执行下列步骤:

- a) 选取一个 t 作为安全参数。
- b) 选取一个大素数阶为 p 的非对称的双线性群组对 (G_1, G_2) , 并且满足一个双线性函数 $e: G_1 \times G_2 \rightarrow G_T$ 。
- c) 选取 G_1 的生成元 P_1 。
- d) 选取 G_2 的生成元 P_2 。
- e) 选取四个密码杂凑函数 $H_1: \{0, 1\}^* \rightarrow G_1, H_2: \{0, 1\}^* \rightarrow Z_p, H_3: \{0, 1\}^* \rightarrow Z_p, H_4: \{0, 1\}^* \rightarrow Z_p$ 。如何构建该密码杂凑函数的示例在附录 B 中给出。
- f) 选取 Z_p 中的两个随机元素 x, y 。
- g) 计算 $X = [x]P_2$ 和 $Y = [y]P_2$ 。
- h) 输出:
 群公共参数: $(G_1, G_2, G_T, e, P_1, P_2, p, H_1, H_2, H_3, H_4)$,
 群组公钥: (X, Y) ,
 群组成员发布密钥: (x, y) 。

注: 推荐参数的示例参见附录 C 中的 C.2。

6.5.2.3 群组成员发布过程需要在主签名方和发布方之间建立一个安全的鉴别通信信道, 如何建立该信道不在本机制的范围之内规定, 对于每一个签名方, 群组成员发布方产生群组成员签名过程如下:

- a) 群组成员发布方选取随机数 $n_1 \in \{0, 1\}^t$ 。
- b) 群组成员发布方发送 n_1 给主签名方。
- c) 主签名方从 Z_p 中随机选取一个群组成员私钥 f 或从它的种子密钥值导出 f 。
- d) 主签名方计算 $Q_2 = [f]P_1$ 。
- e) 主签名方从 Z_p 选取 u 并计算 $U = [u]P_1$ 。
- f) 主签名方计算 $v = H_2(P_1 \parallel Q_2 \parallel U \parallel X \parallel Y \parallel n_1)$ 。
- g) 主签名方计算 $w = (u + v \cdot f) \bmod p$ 。
- h) 主签名方发送 (Q_2, v, w) 给群组成员发布方。
- i) 群组成员发布方计算 $U' = [w]P_1 - [v]Q_2$ 。
- j) 群组成员发布方计算 $v' = H_2(P_1 \parallel Q_2 \parallel U' \parallel X \parallel Y \parallel n_1)$ 。
- k) 群组成员发布方验证 $v = v'$ 。如果验证失败, 放弃群组成员发布过程。
- l) 群组成员发布方从 Z_p 随机选取整数 r 。
- m) 群组成员发布方计算 $A = [r]P_1, B = [y]A, C = [x]A + [rxy]Q_2$ 。
- n) 群组成员发布方将 (A, B, C) 作为签名方的群组成员证书并发送给主签名方。
- o) 主签名方计算 $D = [f]B$ 。
- p) 主签名方发送 (A, B, C, D) 给辅助签名方。
- q) 辅助签名方验证 $e(A, Y) = e(B, P_2)$ 。
- r) 辅助签名方验证 $e(A + D, X) = e(C, P_2)$ 。
- s) 如果验证失败, 辅助签名方丢弃。
- t) 签名方的群组成员签名密钥为 (f, A, B, C) 。

6.5.3 签名过程

输入群组公钥 (X, Y) , 群组成员签名密钥 (f, A, B, C) 、连接基 bsn 、随机数 $n_V \in \{0, 1\}^t$ 和未签消息 $m \in \{0, 1\}^*$, 其中, 连接基既可以是特殊的字符 $\perp$ 也可以是任意的字符串并由其来实现连接能力, 签名过程执行以下步骤:

注 1: 随机数 n_V 通常由验证方选取。

注 2: 另外一种处理 n_V 的方法是将 n_V 作为消息 m 的一部分。

- a) 主签名方拥有群组成员私钥 f , 同时辅助签名方拥有 (A, B, C, D) ;
- b) 如果 $bsn = \perp$, 从 G_1 中选取随机元素 J , 计算 $J = H_1(bsn)$;
- c) 辅助签名方从 Z_p 中选取随机整数 l ;
- d) 辅助签名方计算 $R = [l]A, S = [l]B, T = [l]C$ 和 $W = [l]D$;
- e) 辅助签名方计算 $c = H_3(R \parallel S \parallel T \parallel W \parallel n_V)$;
- f) 辅助签名方发送 (c, J, S, m, bsn) 给主签名方;
- g) 主签名方计算 $K = [f]J$;
- h) 主签名方选取 $n_T \in \{0, 1\}^t$;
- i) 主签名方 Z_p 中选取随机整数 r ;
- j) 主签名方计算 $R_1 = [r]J$ 和 $R_2 = [r]S$;
- k) 主签名方计算 $h = H_4(c \parallel m \parallel J \parallel K \parallel bsn \parallel R_1 \parallel R_2 \parallel n_T)$;
- l) 主签名方计算 $s = (r + h \cdot f) \bmod p$;
- m) 主签名方发送 (K, h, s, n_T) 给辅助签名方;
- n) 辅助签名方输出该匿名签名 $\sigma = (R, S, T, W, J, K, h, s, n_V, n_T)$ 。

6.5.4 验证过程

输入消息 m 、连接基 bsn 、随机数 $n_V \in \{0, 1\}^t$ 、签名 $(R, S, T, W, J, K, h, s, n_V, n_T)$ 和群组公钥 (X, Y) , 验证过程执行下列步骤:

- a) 如果 $bsn \neq \perp$, 验证 $J = H_1(bsn)$;
- b) 验证 $e(R, Y) = e(S, P_2)$ 和 $e(R + W, X) = e(T, P_2)$;
- c) 计算 $R_1 = [s]J - [h]K$;
- d) 计算 $R_2 = [s]S - [h]W$;
- e) 验证 $h = H_4(H_3(R \parallel S \parallel T \parallel W \parallel n_V) \parallel m \parallel J \parallel K \parallel bsn \parallel R_1 \parallel R_2 \parallel n_T)$;
- f) 撤销检查过程是可选的;
- g) 验证 $R \neq O_E$;
- h) 如果以上的任意一项验证失败, 输出 0(无效), 否则输出 1(有效)。

6.5.5 连接过程

给定两个签名 $\sigma = (R, S, T, W, J, K, h, s, n_V, n_T)$ 和 $\sigma' = (R', S', T', W', J', K', h', s', n_V', n_T')$, 连接过程执行下列步骤:

如果 $J = J'$ 且 $K = K'$, 输出 1(连接), 否则输出 0(非连接)。

注: 如果由于 $J \neq J'$ 连接过程输出 0, 意味着连接过程不能确定是否两个签名由同一个群组成员创建。

6.5.6 撤销过程

该机制撤销过程的细节参见参考文献[10], 在该机制中支持四个类型的撤销(私钥撤销、验证方黑名单撤销、签名撤销和证书更新), 前三个撤销同 6.4.6, 证书更新撤销过程在后面有描述。私钥撤销和

签名撤销既可以是全局撤销也可以是本地撤销,验证方黑名单撤销属于全局撤销,证书更新属于全局撤销。

更新群组公钥过程:

- a) 群组成员发布方从 Z_p 中随机选取 x' ;
- b) 群组成员发布方计算 $X' = [x']P_2$;
- c) 新的群组公钥为 (X', Y) ;
- d) 新的群组成员发布密钥为 (x', y) 。

更新群组证书过程:

- a) 群组成员发布方计算 $\beta = x'/x \bmod p$;
- b) 每一个合法的成员拥有群组成员证书 (A, B, C) :
 - 1) 发布方计算 $C' = [\beta]C$ 并发送 C' 给成员;
 - 2) 成员更新它的证书如 (A, B, C') 。

7 具有打开功能的机制

7.1 概述

本章规定了两种拥有打开能力的数字签名机制。该数字签名的类型称为群组签名。在该类机制中,存在一个群组成员打开方实体,它由签名方委任并可以从一个群组签名中识别出签名方的身份。这两种机制均由密钥生成过程、签名过程、验证过程、打开过程和撤销过程组成。

注: 7.2 和 7.3 所规定的机制及关联的安全性证明基于参考文献[17]和[14]。

7.2 机制 5

7.2.1 符号

下列符号适用于本机制。

- K_n, K, K_c, K_s : 安全参数;
- p_1', p_2' : 素数;
- p_1, p_2 : 长度为 $K_n/2$ 比特的素数, 满足 $p_1 = 2p_1' + 1, p_2 = 2p_2' + 1$;
- n : 长度为 K_n 比特的整数, 满足 $n = p_1 p_2$;
- a_0, a_1, a_2, b, w : 模 n 的二次剩余的元素;
- G : 满足 DDH 假设的一个群组;
- q : G 的阶;
- g : G 的一个生成元;
- y_1, y_2 : Z_q 中的元素;
- Y_1, Y_2 : G 中的元素。

7.2.2 密钥产生过程

7.2.2.1 密钥产生过程由四部分组成: 建立过程、群组成员发布方建立过程、群组成员打开方建立过程和群组成员发布过程, 其中建立过程输出由实体间协商的群组公共参数。群组成员发布方建立过程输出群组公钥和群组发布密钥(简称发布密钥)。群组成员打开方建立过程输出一个群组成员打开方公钥和群组成员打开密钥。群组成员发布过程是通过一个运行在群组成员发布方和群成员之间的交互协议来产生一个的用户的群组成员签名密钥。

7.2.2.2 建立过程输出下列群组公共参数:

- a) 选取下列参数: $K_n, K, K_c, K_s, K_e, K_e'$;
- b) 选取一个密码杂凑函数 $H: \{0,1\}^* \rightarrow \{0,1\}^{K_c}$ 。

注: 推荐参数的示例参见附录 C 中的 C.2。

7.2.2.3 群组成员发布方建立过程由群组成员发布方执行下列步骤:

- a) 选取一个 RSA 模块 $n = p_1 p_2$ 使得 $p_1 = 2p_1' + 1, p_2 = 2p_2' + 1$, 其中 p_1', p_2', p_1, p_2 都是素数并且 n 为 K_n 比特;
- b) 随机选取元素 $a_0, a_1, a_2, b, w \in \text{QR}(n)$;
- c) 输出:
群组公钥(gpk): (n, a_0, a_1, a_2, b, w) ,
群组成员发布密钥: $=(p_1, p_2)$ 。

7.2.2.4 群组成员打开方建立过程由群组成员打开方执行下列步骤:

- a) 选取一个满足 DDH 假设并且阶为 q 的一个群组 G ;
- b) 随机选取一个群组 G 的生成元 g ;
- c) 随机选取元素: $y_1, y_2 \in Z_q$;
- d) 计算 $Y_1 = [y_1]g, Y_2 = [y_2]g$;
- e) 输出:
群组成员打开方公钥(opk): (q, g, Y_1, Y_2) ,
群组成员打开密钥: $=(y_1, y_2)$ 。

7.2.2.5 群组成员发布过程在群组成员发布方(简称发布方)和用户之间 U_i 执行下列步骤:

- a) 用户 U_i 随机选取 $x_i' \in \Lambda$, 其中 Λ 是 $(0, 2^\lambda)$ 的区间, $\lambda = K_n + K + K_s$ 。
- b) 用户 U_i 计算 $C = a_1^{x_i'} \bmod n$ 并与一个正确生成的证据一起发送; 参见附录 F 中 F.1 给出生成证据的证明。
- c) 发布方验证该证据。
- d) 发布方随机选取随机数 $x_i'' \in \Lambda$ 并发送给用户 U_i 。
- e) 用户 U_i 验证 $x_i'' \in \Lambda$ 。
- f) 用户 U_i 计算 $x_i = (x_i' + x_i'') \bmod 2^\lambda$ 和 $(A_i', h_i) = (a_1^{x_i} \bmod n, [x_i]g)$, 用户 U_i 将它与正确生成的证据一起发送。参见附录 F 中 F.1 给出生成证据的证明。
- g) 发布方验证该证据。
- h) 发布方选取一个素数 $e_i' \in \{0,1\}^{K_e'}$, 其中 $e_i = 2^{K_e-1} + e_i'$ 也是素数。
- i) 发布方计算 $A_i = (a_0 A_i')^{1/e_i} \bmod n$ 和 $B_i = b^{1/e_i'} \bmod n$ 。
- j) 发布方存储 $(i, (A_i, e_i', B_i, h_i))$ 到成员列表 LIST。
- k) 发布方发送 (A_i, e_i', B_i) 给用户 U_i 。
- l) 用户 U_i 计算 $e_i = 2^{K_e-1} + e_i'$ 并验证 e_i' 和 e_i 是素数。
- m) 用户 U_i 验证 $a_0 a_1^{x_i} \equiv A_i^{e_i}$ 和 $b \equiv B_i^{e_i'} \bmod n$ 。
- n) 用户 U_i 存储 (A_i, e_i', B_i, h_i) 作为它的群组成员证书并且 x_i 作为它的群组成员私钥。 U_i 的群组成员签名密钥为 $(x_i, A_i, e_i', B_i, h_i)$ 。

7.2.3 签名过程

输入群组公钥 $\text{gpk} = (n, a_0, a_1, a_2, b, w)$ 、群组成员证书 (A_i, e_i', B_i, h_i) 、群组成员私钥 x_i 、由验证方认可的群组成员打开方公钥 $\text{opk} = (q, g, Y_1, Y_2)$ 以及未签消息 $M \in \{0,1\}^*$, 签名过程由签名方 U_i 执行下列步骤:

- a) 选取 $\rho_E \in Z_q$ 并计算 $E = (E_0, E_1, E_2) = ([\rho_E]g, h_i + [\rho_E]Y_1, h_i + [\rho_E]Y_2)$ 。
- b) 随机选取 $\rho_m \in \{0,1\}^{K_n/2}$ 并计算 $A_{\text{COM}} = A_i a_2^{\rho_m} \bmod n$ 和 $s = e_i \rho_m$, 其中 $e_i = 2^{K_e-1} + e_i'$ 。

- c) 随机选取 $\rho_r \in \{0, 1\}^{K_n/2}$ 并计算 $B_{\text{COM}} = B_i \omega^{\rho_r} \bmod n$ 和 $t = e_i' \rho_r$ 。
- d) 随机选取 $\mu_x \in \{0, 1\}^{\lambda + K_c + K_s}$, $\mu_s \in \{0, 1\}^{K_e + (K_n/2) + K_c + K_s}$, $\mu_{e'} \in \{0, 1\}^{K_{e'} + K_c + K_s}$, $\mu_t \in \{0, 1\}^{K_{e'} + (K_n/2) + K_c + K_s}$, 和 $\mu_E \in Z_q$ 。
- e) 计算 $V_{\text{ComCipher}} = (V_{\text{ComCipher0}}, V_{\text{ComCipher1}}, V_{\text{ComCipher2}}) = ([\mu_E]g, [\mu_x]g + [\mu_E]Y_1, [\mu_x]g + [\mu_E]Y_2)$ 。
- f) 计算 $V_{\text{ComMPK}} = a_1^{\mu_x} a_2^{\mu_s} A_{\text{COM}}^{-\mu_{e'}} \bmod n$ 。
- g) 计算 $V_{\text{ComRev}} = \omega^{\mu_t} B_{\text{COM}}^{-\mu_{e'}} \bmod n$ 。
- h) 计算 $c = H(K_n \parallel K_e \parallel K_{e'} \parallel K \parallel K_c \parallel K_s \parallel \text{gpk} \parallel \text{opk} \parallel E \parallel A_{\text{COM}} \parallel B_{\text{COM}} \parallel V_{\text{ComCipher}} \parallel V_{\text{ComMPK}} \parallel V_{\text{ComRev}} \parallel M)$ 。
- i) 计算 $\tau_x = c x_i + \mu_x$, $\tau_s = c s + \mu_s$, $\tau_t = c t + \mu_t$, $\tau_{e'} = c e_i' + \mu_{e'}$, $\tau_E = c \rho_E + \mu_E \bmod q$ 。
- j) 验证 $|\tau_x| \leq \lambda + K_c + K_s$ 和 $|\tau_{e'}| \leq K_{e'} + K_c + K_s$ 。如果它们都不成立, 返回步骤 d)。
- k) 输出 $\sigma = (E, A_{\text{COM}}, B_{\text{COM}}, c, \tau_x, \tau_s, \tau_{e'}, \tau_t, \tau_E)$ 作为对消息 M 的签名。

7.2.4 验证过程

当输入消息 M , 签名 $\sigma = (E, A_{\text{COM}}, B_{\text{COM}}, c, \tau_x, \tau_s, \tau_{e'}, \tau_t, \tau_E)$ 、群组公钥 $\text{gpk} = (n, a_0, a_1, a_2, b, \omega)$ 以及群组成员打开方公钥 $\text{opk} = (q, g, Y_1, Y_2)$, 验证过程执行下列步骤:

- a) 验证 $|\tau_x| \leq \lambda + K_c + K_s$ 和 $|\tau_{e'}| \leq K_{e'} + K_c + K_s$;
- b) 计算 $\tau_e = \tau_{e'} + c \cdot 2^{K_e} - 1$, $V'_{\text{ComCipher}} = (V'_{\text{ComCipher0}}, V'_{\text{ComCipher1}}, V'_{\text{ComCipher2}}) = ([\tau_E]g - [c]E_0, [\tau_x]g + [\tau_E]Y_1 - [c]E_1, [\tau_x]g + [\tau_E]Y_2 - [c]E_2)$, $V'_{\text{ComMPK}} = a_0^c a_1^{\tau_x} a_2^{\tau_s} A_{\text{COM}}^{-\tau_e} \bmod n$ 和 $V'_{\text{ComRev}} = b^c \omega^{\tau_t} B_{\text{COM}}^{-\tau_{e'}} \bmod n$;
- c) 验证 $c = H(K_n \parallel K_e \parallel K_{e'} \parallel K \parallel K_c \parallel K_s \parallel \text{gpk} \parallel \text{opk} \parallel E \parallel A_{\text{COM}} \parallel B_{\text{COM}} \parallel V'_{\text{ComCipher}} \parallel V'_{\text{ComMPK}} \parallel V'_{\text{ComRev}} \parallel M)$;
- d) 如果以上的任意一种验证失败, 输出 0(无效), 否则输出 1(有效)。

7.2.5 打开过程

给定签名 $\sigma = (E, A_{\text{COM}}, B_{\text{COM}}, c, \tau_x, \tau_s, \tau_{e'}, \tau_t, \tau_E)$, 群组成员打开过程需要通过群组成员打开方使用群组成员打开密钥 (y_1, y_2) 执行下列步骤:

- a) 计算 $S_1 = E_1 - [y_1]E_0$ 和 $S_2 = E_2 - [y_2]E_0$ 。验证 $S_1 = S_2$ 并且设置 $h = S_1$;
- b) 在成员列表 LIST 内搜索 h 并输出对应的用户 ID;
- c) 否则输出打开失败。

7.2.6 撤销过程

当用户准备离开该群组或使他的成员资格被撤销, 群组成员发布方需要更新群组公钥, 并且其他的用户为了能够用新的群组公钥继续产生签名需要去更新他的成员证书。该撤销过程属于全局撤销。

更新群组公钥的过程:

设 $\text{mpk}' = (A, e, B, h)$ 是一个要离开的用户的公钥, 群组成员撤销过程需要通过群组成员发布方使用群组成员发布密钥 (p_1, p_2) 执行下列步骤:

- a) 计算 $b' = b^{1/e} \bmod n$;
- b) 更新群组公钥 gpk 为 $(n, a_0, a_1, a_2, b', \omega)$ 和撤销列表 RL 为 $\text{RL} \cup \{(\text{mpk}', \text{gpk})\}$ 。

更新成员凭证的过程:

在撤销列表 RL 中, 已知 $\text{mpk}_i = (A_i, e_i', h_i, B_i)$, $(\text{mpk}' = (A, e, B, h), \text{gpk} = (n, a_0, a_1, a_2, b', \omega))$ 以及更新群组成员凭证过程的 mpk_i , 执行下列步骤:

- a) 计算 α, β 使得 $\alpha e + \beta e_i' = 1$;
- b) 计算 $B_i' = B_i^{\alpha} b^{\beta} \bmod n$ 并用 (A_i, e_i', B_i', h_i) 替换 mpk_i 。

7.3 机制 6

7.3.1 符号

下列符号适用于本机制。

- $P_1, Q_1, R_1, U, U', A, B; G_1$ 中的元素；
- $P_2, Y; G_2$ 中的元素；
- $P_3, S, T, W, W', Z, V, Z', V', W'; G_3$ 中的元素；
- $Y'; G_T$ 中的元素；
- $x, s, t, f, c', c, d, f', u', r, q, y, v, z, c, y', z', h', g', u, h, g, com, a, b, j, n, o; Z_p$ 中的整数；
- H : 输出元素在 Z_p 内的密码杂凑函数。

7.3.2 密钥产生过程

7.3.2.1 密钥产生过程由两部分组成：建立过程和群组成员发布过程，其中建立过程是群组成员发布方产生群组公共参数、群组公钥和群组发布密钥的过程。群组成员发布过程是通过一个运行在群成员发布方和群成员之间的交互协议来产生唯一的群组成员签名。

7.3.2.2 该建立过程由群组成员发布方执行下列步骤：

- a) 选取一个阶为 p 的非对称双线性大素数群组对并且其对应的双线性函数为 $e: G_1 \times G_2 \rightarrow G_T$ 。
- b) 选取一个阶为 p 的素数的循环群 G_3 满足 DDH 难解性问题。如果 G_3 是一个椭圆曲线，它的域的特征应该不同于 G_1 和 G_2 所属域的特征。
- c) 随机选取 G_1 的生成元 P_1, Q_1, R_1 。
- d) 随机选取 G_2 的生成元 P_2 。
- e) 随机选取 G_3 的生成元 P_3 。
- f) 选取一个密码杂凑函数 $H: \{0, 1\}^* \rightarrow Z_p$ 。在附录 B 中给出如何构建该密码杂凑函数的示例。
- g) 在 Z_p 中选取三个随机整数 x, s, t 。
- h) 计算 $X = [x]P_2, S = [s]P_3$ 和 $T = [t]P_3$ 。
- i) 输出：群组公共参数： $= (G_1, G_2, G_T, e, G_3, p, H)$ ，
群组公钥： $= (P_1, Q_1, R_1, P_2, P_3, X, S, T)$ ，
打开密钥： $= (s, t)$ ，
群组成员发布密钥： $= (x)$ 。

注：推荐参数的示例参见附录 C 中的 C.2。

7.3.2.3 群组成员发布过程需要在主签名方和发布方之间建立一个安全的鉴别通信信道，如何建立该信道不在本机制的范围之内规定，群组成员发布过程包括以下步骤：

- a) 签名方从 Z_p 中随机选取 f 或从它的种子密钥值导出，其中 f 是群组成员私钥的一部分。
- b) 签名方计算 $W = [f]P_3$ 。
- c) 签名方从 Z_p 中选取 u 并计算 $U = [f]Q_1 + [u]R_1$ 。
- d) 签名方发送 (W, U) 给群组成员发布方。
- e) 群组成员发布方从 Z_p 中随机选取 c' 和 d 并计算 $com = H(c' \parallel d)$ 。
- f) 群组成员发布方发送 com 给签名方。
- g) 签名方随机选择 f' 和 u' 并计算 $W' = [f']P_3$ 和 $U' = [f']Q_1 + [u']R_1$ 。
- h) 签名方发送 (W', U') 给群组成员发布方。
- i) 群组成员发布方发送 (c', d) 给签名方。
- j) 签名方验证 $com = H(c' \parallel d)$ 。如果验证失败，终止群成员发布过程。

- k) 签名方计算 $r = fc' + f' \bmod p$ 和 $q = uc' + u' \bmod p$ 。
- l) 签名方发送 (r, q) 给群组成员发布方。
- m) 群组成员发布方验证 $[r]P_3 = [c']W + W'$ 和 $[r]Q_1 + [q]R_1 = [c']U + U'$ ，如果验证失败，终止群成员发布过程。
- n) 群组成员发布方从 Z_p 中选择随机整数 y 和 v 。
- o) 群组成员发布方计算 $A = [1/(x+y)](P_1 - U - [v]R_1)$ 。
- p) 群组成员发布方将 (A, y) 作为签名方的群组成员凭证并发送给签名方。
- q) 签名方计算 $z = u + v$ ，并将 (A, y) 作为群组成员凭证。
- r) 签名方验证 $e(A, X + [y]P_2) \cdot e([f]Q_1, P_2) \cdot e([z]R_1, P_2) = e(P_1, P_2)$ 。
- s) 如果验证失败，签名方退出。
- t) 签名方的群组成员签名密钥为 (f, A, y, z) 。

7.3.3 签名过程

输入群组公钥 $(P_1, Q_1, R_1, P_2, P_3, X, S, T)$ 、群组成员签名密钥 (f, A, y, z) 和未签消息 $m \in \{0, 1\}^*$ ，该签名过程执行下列步骤：

- a) 签名方拥有群组成员签名密钥 (f, A, y, z) ；
- b) 签名方从 Z_p 中随机选取 g 和 h ；
- c) 签名方计算 $B = A + [h]R_1$ ， $Z = [f + g]P_3$ ， $V = [g]S$ 和 $W = [g]T$ ；
- d) 签名方从 Z_p 中随机选取 a, b, j, n 和 o ；
- e) 签名方计算 $Y = e(Q_1, P_2)^a \cdot e(B, P_2)^b \cdot e(R_1, P_2)^j \cdot e(R_1, X)^n$ ， $Z' = [a + o]P_3$ ， $V' = [o]P_3$ 和 $W' = [o]P_3$ ；
- f) 签名方计算 $c = H(p \parallel P_1 \parallel P_2 \parallel P_3 \parallel X \parallel S \parallel T \parallel Q_1 \parallel R_1 \parallel B \parallel Z \parallel V \parallel W \parallel Y \parallel V' \parallel W' \parallel Z' \parallel m)$ ；
- g) 签名方计算 $f' = cf + a \bmod p$ ， $y' = cy + b \bmod p$ ， $z' = c(z - hy) + j \bmod p$ ， $h' = -ch + n \bmod p$ 和 $g' = cg + o \bmod p$ ；
- h) 签名方输出群组签名 $\sigma = (B, Z, V, W, c, f', y', z', h', g')$ 。

7.3.4 验证过程

输入消息 m 、签名 $\sigma = (B, Z, V, W, c, f', y', z', h', g')$ 以及群组公钥 $(P_1, Q_1, R_1, P_2, P_3, X, S, T)$ ，验证过程执行下列步骤：

- a) 计算 $Y = e(Q_1, P_2)^{f'} \cdot e(B, [y']P_2 + [c]X) \cdot e(R_1, P_2)^{z'} \cdot e(R_1, X)^{h'} \cdot e(P_1, P_2)^{-c}$ ， $Z' = [f' + g']P_3 - [c]Z$ ， $V' = [g']S - [c]V$ 和 $W' = [g']T - [c]W$ ；
- b) 验证 $c = H(p \parallel P_1 \parallel P_2 \parallel P_3 \parallel X \parallel S \parallel T \parallel Q_1 \parallel R_1 \parallel B \parallel Z \parallel V \parallel W \parallel Y \parallel V' \parallel W' \parallel Z' \parallel m)$ 成立；
- c) 如果以上验证失败，输出 0（有效），否则输出 1（无效）。

7.3.5 打开过程

给定签名 $\sigma = (B, Z, V, W, c, f', y', z', h', g')$ ，打开过程执行下列步骤：

- a) 如果验证失败，输出 $\perp$ （失败）；
- b) 计算 $W = Z - [1/s]V$ ；
- c) 输出 W 。

7.3.6 撤销过程

该撤销过程属于全局撤销，过程如下：

- a) 群组成员发布方收到一个已撤销的签名方群组成员证书($\underline{A}, \underline{y}$);
- b) 群组成员发布方计算 $\underline{Q}_1 = [1/(x+y)]Q_1$, $\underline{P}_1 = [1/(x+y)]P_1$ 和 $\underline{R}_1 = [1/(x+y)]R_1$;
- c) 群组成员发布方发送($\underline{y}, \underline{Q}_1, \underline{P}_1, \underline{R}_1$) 给每一个签名方;
- d) 群组成员签名密钥是(f, A, y, z)的每一个签名方, 计算 $A' = [1/(\underline{y}-y)](A - \underline{P}_1 - [f]Q_1 - [z]R_1)$ 并设它的群组签名密钥为(f, A', y, z);
- e) 每一个签名方设置群组公钥为($\underline{P}_1, \underline{Q}_1, \underline{R}_1, P_2, P_3, X, S, T$)。

8 具有打开和连接功能的机制

8.1 概述

该部分给出了两个同时拥有打开功能和连接功能的数字签名机制。该类数字签名也可称为带控制连接能力的群组签名。在该类机制中, 有一个实体称为群组成员打开方, 它可以由签名方制定并且可以从群组签名中验证签名方的身份。还有一个实体称为群组签名连接方, 它可以确定两个签名是否由同一个群组成员创建。机制 8 用于 GB/T 34953.2—2018 中的匿名实体鉴别机制。

注: 8.2 所规定的机制及关联的安全性证明基于参考文献[15]和[16]。

8.2 机制 7

8.2.1 符号

下列符号适用于本机制。

- $Q, Q_1, Q_2, U, W, D, V, A, Z, W_{ID}, Q_1', Q_2', U', W', D', \tilde{A}, D_1, D_2, D_3, R_1, R_2, R_3, K_{open}, W_{open}, V_{open}, Y_{1,i}, Y_{2,j}, X_{1,i}, X_{2,i}, S_{1,j}, S_{2,j}, S_{3,j}, S_{4,j}, S_{5,j}$: G_1 中的元素;
- B_1, B_θ : G_2 中的元素;
- $L_1, L_2, L_3, L_4, L_A, L_1', L_2', L_3', L_4'$: G_T 中的元素;
- $\eta, \xi, \theta, x, y, z, r_{ID}, c_{ID}, s_{ID}, \alpha, \gamma, r_a, r_x, r_y, r_y, c, s_a, s_x, s_y, c_{open}, s_{open}, x_1, x_2, x_j, v_j$: Z_p^* 中的整数;
- $i, j, \lambda, \rho, \kappa$: 长度为 t 比特的整数, 这里 t 是一个固定的非负整数;
- H_p : 输出元素在 Z_p^* 内的密码杂凑函数。

该机制由密钥产生过程(包括建立过程和群组成员发布过程)、签名过程、验证过程、打开过程、连接过程和撤销过程。证据评估过程用来验证打开过程产生的绑定证据的合法性。群组签名撤销过程包括三个子过程: 产生 RL、更新 gpk 和更新 usk 。

8.2.2 密钥产生过程

8.2.2.1 密钥产生过程由两部分组成: 建立过程和群组成员发布过程。建立过程输入安全参数并产生群组公钥 gpk 和它对应的群组成员发布密钥 $gmik$, 群组成员打开密钥 $gmok$ 和群组签名连接密钥 $gslk$ 。

8.2.2.2 产生群组公共参数过程如下:

- a) 产生三个阶为 p 的素数群组 G_1, G_2, G_T 和一个双线性映射 $e: G_1 \times G_2 \rightarrow G_T$ 。假设群组是乘法群。
- b) 选取 $B_1 \leftarrow G_2$ 和 $Q_1, Q_2, Q, U \leftarrow G_1$ 。
- c) 选取一个密码杂凑函数 $H_p: \{0, 1\}^* \rightarrow Z_p^*$, 其中 $H_p(M)$ 是消息 $M \in \{0, 1\}^*$ 的散列码。在附录 B 中给出了一个如何构建该密码杂凑函数的示例。

8.2.2.3 密钥建立过程如下:

- a) 选取 $\eta, \xi, \theta \leftarrow Z_p^*$;
- b) 计算 $W = [\eta]U, D = [\xi]U, B_\theta = [\theta]B_1, V = [\xi]B_1$;
- c) 计算 $L_1 = e(W, B_1), L_2 = e(W, B_\theta), L_3 = e(Q_1, B_1)$ 和 $L_4 = e(Q_2, B_1)$;
- d) 输出群组公共参数: $((e, G_1, G_2, G_T), Q, B_1, B_\theta, H_p)$;
- e) 输出初始群组公钥: $gpk = (Q_1, Q_2, U, W, D, (L_1, L_2, L_3, L_4))$;
- f) 输出: $gmik = \theta, gmok = (\eta, \xi)$ 和 $gslk = (V)$ 。

注 1: L_1, L_2, L_3 和 L_4 在 gpk 内是可选的, 因为他们可以由签名方和验证方通过 $Q_1, Q_2, B_1, B_\theta, W$ 计算得出。

注 2: 推荐参数的示例参见附录 C 中的 C.2。

管理密钥 $gmik, gmok$ 和 $gslk$ 分别由群组成员发布方、群组成员打开方和群组签名连接方安全存储。

8.2.2.4 群组成员发布过程如下:

群组成员发布方管理一个成员列表 $LIST = (LIST[1], \dots, LIST[n])$, 其中, n 是目前已注册群组成员的数量。列表的每一条记录包含每一个注册用户的个人信息。

群组成员发布包括两个子过程: 用户加入子过程(由加入用户使用其身份 ID 进行)和发布过程(由群组成员发布方运行)。假设两个子过程通过一个安全的鉴别信道, 产生一个群组签名密钥过程如下:

用户加入过程:

- a) 选取随机群组成员私钥 $sk = z \leftarrow Z_p^*$ 并计算 $Z = [z]W$;
- b) 选取 $r_{ID} \leftarrow Z_p^*$ 并计算 $W_{ID} = [r_{ID}]W$;
- c) 选取 $c_{ID} = H_p(ID \parallel W \parallel Z \parallel W_{ID})$;
- d) 计算 $s_{ID} = r_{ID} + c_{ID}z \pmod{p}$;
- e) $T_{ID} = (Z, s_{ID}, c_{ID})$;
- f) 发送 $(Join_Request, ID, T_{ID})$ 给发布过程。

注: 在发布过程结束后执行用户加入过程(续)的步骤 g)。

发布过程:

- a') 接收一个加入请求消息 $(Join_Request, ID, T_{ID})$ 。
- b') 检查消息的合法性, 过程如下:
 - 1) 检查 ID 是否合法;
 - 2) 检查是否满足 $c_{ID} = H_p(ID \parallel W \parallel Z \parallel [s_{ID}]W - [c_{ID}]Z)$ 。
- c') 检查 $LIST[i] = (ID, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot)$ 中的 ID 是否包含在成员列表 $LIST = (LIST[1], \dots, LIST[n])$ 中。
- d') 如果 ID 没有被注册, 例如, 没有匹配的 i 存在, 那么进程如下:
 - 1) 选取 $x, y \leftarrow Z_p^*$;
 - 2) 计算 $A = [1/(\theta + x)](Q_1 - [y]Q_2 - Z) (= [1/(\theta + x)](Q_1 - [y]Q_2 - [z]W))$;
 - 3) 增加 $LIST[n+1] = (ID, [y]Q, A, x, y, upk[i] = Z (= [z]W), T_{ID})$ 到 $LIST$;
- 否则:
 - 1) 选取 $x \leftarrow Z_p^*$;
 - 2) 计算 $A = [1/(\theta + x)](Q_1 - [y]Q_2 - Z) (= [1/(\theta + x)](Q_1 - [y]Q_2 - [z]W))$;
 - 3) 用新的 $LIST[i] = (ID, [y]Q, A, x, y, upk[i] = Z, T_U)$ 代替之前的 $LIST[i]$ 。
- e') 发送群组成员证书 (i, A, x, y) 到用户加入算法;

用户加入过程(续):

注: 接续发布过程前的用户加入过程的步骤 f)。

- g) 接收消息 (i, A, x, y) 。
- h) 验证是否满足 $e(A, B_\theta + [x]B_1) = e(Q_1 - [y]Q_2 - [z]W, B_1)$, 其中, Q_1, Q_2, W 包含在 gpk

内。如果等式不成立,则终止。否则, i^{th} 群组成员签名密钥 $usk_{i0} = (0, x, y, z, A = [1/(\theta + x)](Q_1 - [y]Q_2 - [z]W))$ 。值 0 意味着该密钥是由群组成员发布方发布的初始群组成员签名密钥,该密钥将通过撤销更新并且 0 将根据更新变为相应的指数。

8.2.3 签名过程

群组成员签名密钥包括一个指数 κ , κ 表示该密钥已经进行了 κ 次撤销列表指数 RI(群组签名撤销过程相关)注册更新。设 $\lambda (\geq \kappa)$ 是 RI 中最新的撤销密钥数量。为了产生一个签名,群组成员签名密钥需要 λ 次 RI 的注册更新。已产生的签名包括一个 λ , λ 表示签名是使用已进行 λ 次 RI 的注册更新的密钥生成的。该签名可以使用已进行 λ 次 RI 的注册更新的群组公钥验证。

该签名过程输入群组公共参数 $((e, G_1, G_2, G_T), Q, B_1, B_\theta, H_\rho)$ 、群组公钥 gpk_κ 、群组成员签名密钥 $usk_{i\kappa} = (\kappa, x, y, z, A)$ 以及消息 $M \in \{0, 1\}^*$, 其中 $\kappa (\leq \lambda)$ 是签名方 i 最后更新的撤销列表指数 i , 而 λ 在所有群组成员中最新的撤销列表指数。该过程执行下列步骤:

- 调用 $usk_{i\kappa}$ 更新群组签名撤销过程, 并获取 $gpk_\lambda = (Q_1', Q_2', U', W', D', (L_1', L_2', L_3', L_4'))$ 和一个已更新的群组成员签名密钥 $usk_{i\lambda} = (\lambda, x, y, z, \tilde{A})$, 其中 $\tilde{A}$ 已经是第 λ 次 RI 的注册更新;
- 选取 $\alpha \leftarrow Z_p^*$;
- 计算 $D_1 = [\alpha]U', D_2 = \tilde{A} + [\alpha]W', D_3 = [y]Q + [\alpha]D'$;
- 计算 $\gamma = x\alpha - z \pmod p$;
- 选取 $r_a, r_x, r_\gamma, r_y \leftarrow Z_p^*$;
- 计算 $R_1 = [r_a]U'$;
- 计算 $R_2 = e(D_2, B_1)^{r_x} e(W', B_\theta)^{-r_a} e(W', B_1)^{-r_\gamma} e(Q_2', B_1)^{r_y}$;
- 计算 $R_3 = [r_y]Q + [r_a]D'$;
- 计算 $c = H_\rho(M \parallel \lambda \parallel D_1 \parallel D_2 \parallel D_3 \parallel R_1 \parallel R_2 \parallel R_3)$;
- 计算 $s_a = r_a + c\alpha \pmod p, s_x = r_x + cx \pmod p, s_\gamma = r_\gamma + c\gamma \pmod p, s_y = r_y + cy \pmod p$;
- 输出 $\sigma = (\lambda, D_1, D_2, D_3, c, s_a, s_x, s_\gamma, s_y)$ 。

注: 使用预计算 $L_A = e(\tilde{A}, B_1)$, 步骤 g) 可以修改如下: 计算 $R_2 = L_A^{r_x} \cdot (L_1')^{ar_x - r_\gamma} \cdot (L_2')^{-r_a} \cdot (L_4')^{-r_y}$ 。

8.2.4 验证过程

为了验证具有指数为 ρ 的签名, 使用已经过更新到第 ρ 次 RI 的群组公钥, 应注意: 由于一些密钥可能在之前的验证中被撤销, ρ 可能不是 RI 中的最新的撤销密钥数量。

注 1: 群组签名验证过程输入群组公共参数 $((e, G_1, G_2, G_T), Q, B_1, B_\theta, H_\rho)$ 、 $gpk_0 = (Q_1, Q_2, U, W, D, (L_1, L_2, L_3, L_4))$ 、签名 $\sigma = (\rho, D_1, D_2, D_3, c, s_a, s_x, s_\gamma, s_y)$ 和消息 $M \in \{0, 1\}^*$, 其中, ρ 为在签名产生过程中的撤销指数。

该过程执行如下步骤:

- 调用 $gpk, (\rho, RL)$ 更新撤销的 gpk , 并获取 $gpk_\rho = (Q_1', Q_2', U', W', D', (L_1', L_2', L_3', L_4'))$;
- 计算 $R_1 = [s_a]U' - [c]D_1$;
- 计算 $R_2 = e(D_2, B_1)^{s_x} e(W', B_\theta)^{-s_a} e(W', B_1)^{-s_\gamma} e(Q_2', B_1)^{s_y} (e(D_2, B_\theta) / e(Q_1', B_1))^c$;
- 计算 $R_3 = [s_y]Q + [s_a]D' - [c]D_3$;
- 检查等式 $c = H_\rho(M \parallel \rho \parallel D_1 \parallel D_2 \parallel D_3 \parallel R_1 \parallel R_2 \parallel R_3)$ 是否成立;
- 如果等式成立, 则输出 1 (有效); 否则输出 0 (无效)。

注 2: 为了减少双线性计算, 步骤 c) 可以修改为: 计算 $R_2 = e(D_2, [s_x]B_1 + [c]B_\theta) L_2'^{-s_a} L_1'^{-s_\gamma} L_4'^{s_y} L_3'^{-c}$ 。

8.2.5 打开过程

群组成员打开过程输入签名 $\sigma = (\rho, D_1, D_2, D_3, c, s_a, s_x, s_y, s_z)$ 和群组成员打开密钥 $gmok = (\eta, \xi)$, 执行如下步骤:

- a) 计算 $[y]Q = D_3 - [\xi]D_1$ 。
- b) 从成员列表 LIST 找到对应 i 的 $LIST[i] = (ID, [y]Q, A, x, y, upk[i] = Z([z]W), \cdot)$ 。
- c) 如果没有相匹配的 i , 输出 $(i=0, *)$ 。否则, 执行如下过程:
 - 1) 选取 $r \leftarrow Z_p^*$;
 - 2) 计算 $K_{open} = [\eta]D_1, W_{open} = [r]U, V_{open} = [r]D_1, c_{open} = H_p(\sigma \parallel g \parallel K_{open} \parallel W_{open} \parallel V_{open})$ 和 $s_{open} = r + c_{open}\eta \pmod{p}$;
 - 3) 输出一个绑定证据 $(i, \tau = (K_{open}, c_{open}, s_{open}), upk[i] = Z, Y_{1,i} = [y_i]Q_2, Y_{2,i} = [y_i]B_1, X_{1,i} = [x_i]Q, X_{2,i} = [x_i]B_1)$;
 - 4) 如果使用该绑定证据的证据评估过程输出为 1, 输出用户 ID 对应为 i 。

8.2.6 证据评估过程

证据评估过程输入群组公钥 gpk 、签名 $\sigma = (\rho, D_1, D_2, D_3, c, s_a, s_x, s_y, s_z)$ 、绑定证据 $(i, \tau = (K_{open}, c_{open}, s_{open}))$ 和 $upk[i] = Z, Y_{1,i}, Y_{2,i}, X_{1,i}, X_{2,i})$, 则执行过程如下:

- a) 调用 $gpk, (\rho, RL)$ 更新撤销的 gpk , 并获取 $gpk_\rho = (Q_1', Q_2', U', W', D', (L_1', L_2', L_3', L_4'))$;
- b) 如果 $i=0$, 那么输出 $\perp$ (失败)。否则, 检查是否下列等式成立:

$$c_{open} = H_p(\sigma \parallel g \parallel K_{open} \parallel [s_{open}]U - [c_{open}]W \parallel [s_{open}]D_1 - [c_{open}]K_{open}), e(D_2 - K_{open}, X_{2,i} + B_\theta) = e(Q_1 - Y_{1,i} - Z_i, B_1'),$$
 其中, Q_1 包含在 gpk , 并且 $\log_{B_1} B_1' = \log_{Q_1} Q_1'$, 如果以上所有的等式成立, 则输出 1 (有效); 否则输出 0 (无效)。

注: 假设 $e(X_{1,i}, B_1) = e(Q, X_{2,i})$ 和 $e(Y_{1,i}, B_1) = e(Q_2, Y_{2,i})$, 其中 Q 和 Q_2 包含在 gpk 中。

8.2.7 连接过程

群组签名连接过程输入两个签名 σ 和 σ' , 群组签名连接密钥 $gslk = (B_1, V = [\xi]B_1)$, 则执行过程如下:

- a) 计算 $LI_1 = e(D_3, B_1)e(D_1, V)^{-1}$ 和 $LI_2 = e(D_3', B_1)e(D_1', V)^{-1}$;
- b) 如果 $LI_1 = LI_2$, 则输出 1 (连接), 否则输出 0 (非连接)。

注: a) 可以修改如下: 计算 $LI_1 = e(D_3 - D_3', B_1)$ 和 $LI_2 = e(D_1 - D_1', V)$ 。

8.2.8 撤销过程

8.2.8.1 RI 是撤销指数列表, 它包含目前已撤销群组成员的所有指数。无论何时密钥被撤销, RI 都能马上将它更新进去。RL 是一个包含目前已撤销群组成员隐私信息的列表。假设 RL 总是更新到最新的撤销指数 RI。任何人都可以公开的使用 RL 和 RI。列表 RI 和 RL 的初始设置为空。该撤销过程是一个全局撤销。它执行三个子过程: 产生 RL (由发布方执行)、更新 gpk (由任意一方执行) 和更新 usk (由合法签名方或群组成员执行)。

8.2.8.2 产生 RL:

- a) 假设 $RI = \{j_1, \dots, j_\lambda\}$ 已给定, 让 $LIST[i] = (ID, [y_i]Q, A_i, x_i, y_i, \cdot, \cdot)$ 在成员列表 LIST 中。定义 $v_k = (\theta - x_{j_1})(\theta - x_{j_2}) \cdots (\theta - x_{j_k}) \pmod{p}$;
- b) 对于每一个 $j \in RI$, 执行如下动作:
 - 1) 计算 $S_{1,j} = [1/v_j]Q_1, S_{2,j} = [1/v_j]Q_2, S_{3,j} = [1/v_j]U, S_{4,j} = [1/v_j]W, S_{5,j} = [1/v_j]D$;

- 2) 增加 $(S_{1,j}, S_{2,j}, S_{3,j}, S_{4,j}, S_{5,j}, x_j)$ 到 RL。
- c) 公开 $RL = \{(S_{1,j}, S_{2,j}, S_{3,j}, S_{4,j}, S_{5,j}, x_j) | j \in RI = \{j_1, \dots, j_\lambda\}\}$ 。
- 8.2.8.3 更新 gpk 过程如下：**
- 输入初始 gpk , 指数 ρ 和 RL;
 - 从 RL 获取 λ , 其中 λ 是目前所有撤销密钥的数量;
 - 如果 $\lambda < \rho$ 或 $\rho < 0$, 则终止;
 - 否则, 例如 $0 \leq \rho < \lambda$, 则更新群组公钥为第 ρ 个撤销密钥。
 - 设置 $RI(\rho) = \{j_1, \dots, j_\rho\} \subseteq RI$ 和 $RL(\rho) = \{(S_{1,j}, S_{2,j}, S_{3,j}, S_{4,j}, S_{5,j}, x_j) | j \in RI = \{j_1, \dots, j_\rho\}\}$;
 - 计算 $Q_1' = S_{1,j_\rho}, Q_2' = S_{2,j_\rho}, U' = S_{3,j_\rho}, W' = S_{4,j_\rho}$ 和 $D' = S_{5,j_\rho}$;
 - 计算 $L_1' = e(W', B_1), L_2 = e(W', B_\theta), L_3 = e(Q_1', B_1)$ 和 $L_4' = e(Q_2', B_1)$;
 - 输出一个更新的群组公钥 $gpk_\rho = (Q_1', Q_2', U', W', D', (L_1', L_2', L_3', L_4'))$ 。
- 8.2.8.4 更新 usk ：**
- 输入 $usk_{ik} = (\kappa, x, y, z, A)$;
 - 计算 gpk_λ , 通过使用 $(gpk, (-1, RL))$ 更新 gpk ;
 - 令 $RI(\kappa, \lambda) = \{j_{\kappa+1}, \dots, j_\lambda\} \subseteq RI$;
 - 计算 $\tilde{A} = [(-1)^{\lambda-\kappa} / \pi_{\lambda-\kappa}]A + (\Pi_j \subseteq RI_{(\kappa, \lambda)} [(-1)^{\lambda-\kappa-j} \pi_{j-1} / \pi_{\lambda-\kappa}](S_{1,j} + [-y]S_{2,j} + [-z]S_{3,j}))$, 其中, $S_{1,j}, S_{2,j}, S_{3,j}, x_j \in RL, \pi_0 = 1, \pi_j = (x - x_1)(x - x_2) \cdots (x - x_j) \pmod p$;
 - 输出 $(gpk_\lambda, usk_{ik} = (\lambda, x, y, z, \tilde{A}))$ 。

8.3 机制 8

8.3.1 符号

本机制基于 GB/T 32918.2—2016。下列符号适用于本机制。

- l : 安全参数;
- $U, F, J, K, K', A, B, C, D, W, Q, V, R, R_1, R_2, R_1^+, R_2^+, J', A', B', C', D', C_1, C_2, C_3, C_1', C_2', C_3', C^-$: G_1 中的元素;
- X', X, Y : G_2 中的元素;
- $x', e_1, e_2, c_1, s_1, t_1, c_2, a', \theta, f, f', x, y, c, v, w, e, e', s, a, u, r, r', c^+, e^+, e^*, t, t^+$: Z_p 中的整数;
- n_l, n_v, n'_v : 长度为 l 比特的整数;
- H : 输出 G_1 中元素的密码杂凑函数;
- H_1 : 输出 Z_p 中元素的密码杂凑函数;
- n_v, n'_v : 长度为 l 比特的整数。

8.3.2 密钥产生过程

8.3.2.1 密钥产生过程由两部分组成：建立过程和群组成员发布过程。其中, 建立过程是群组成员发布方产生证明、群组公共参数、群组公钥和群组发布密钥的过程; 群组成员发布过程是通过一个运行在群成员发布方和群成员之间的交互协议来产生唯一群组成员签名的过程。

8.3.2.2 密钥建立过程由群组成员发布方通过下列步骤完成：

- 选取非对称双线性群对 (G_1, G_2) , G_1, G_2 的阶数都为 p , 映射函数 $\hat{t}: G_1 \times G_2 \rightarrow G_T$ 。

- b) 选取 G_1 中的随机生成元 P_1 。
- c) 选取 G_2 中的随机生成元 P_2 。
- d) 选取两个密码杂凑函数 $H: \{0,1\}^* \rightarrow G_1, H_1: \{0,1\}^* \rightarrow Z_p$ 。密码杂凑函数的选取见附录 B。
- e) 选取 Z_p 中的随机数 x, y 。
- f) 计算 $X = [x]P_2$ 和 $Y = [y]P_2$ 。
- g) 输出：
 - 1) 群公共参数: $G_1, G_2, G_T, \hat{e}, P_1, P_2, p, H, H_1$;
 - 2) 群公钥: (X, Y) , 即 PK , 可表示为 $X \parallel Y$;
 - 3) 群组成员发布密钥: x, y 。

8.3.2.3 群组成员打开过程的密钥由群成员打开方通过下列步骤完成：

- a) 选取随机数: $\alpha, \beta \in Z_p$;
- b) 选取 G_1 中的一个元素 W , 计算 $Q = [\alpha - 1]W$ 和 $V = [\beta - 1]W$;
- c) 输出：
 - 1) 群组成员打开方公钥: $opk = (Q, V, W)$;
 - 2) 群组成员打开方密钥: α, β 。

8.3.2.4 群组成员发布过程需要在主签名方和发布方之间建立一个安全的鉴别通信信道, 如何建立该信道不在本机制的范围之内, 群组成员发布过程包括下列步骤：

- a) 群组成员发布方选取随机数 $n_I \leftarrow \{0,1\}^l$;
- b) 群组成员发布方发送 n_I 给主签名方;
- c) 主签名方从 Z_p 中随机选取一个群组成员私钥 f , 或者由密钥种子值导出 f ;
- d) 主签名方计算 $F = [f]P_1$;
- e) 主签名方从 Z_p 中选取 u 并计算椭圆曲线上的点 $U = [u]P_1$, U 由 (x_U, y_U) 表示;
- f) 主签名方计算 $e = H_1(PK \parallel P_1 \parallel F \parallel n_I)$;
- g) 主签名方计算 $v = e + x_U \bmod q$ 和 $w = (1 + f)^{-1} \cdot (u - v \cdot f) \bmod q$;
- h) 主签名方发送 (F, v, w) 给群组成员发布方;
- i) 群组成员发布方计算 $t = v + w \bmod q$;
- j) 群组成员发布方验证是否满足 $t \neq 0$, 如果 $t = 0$ 即验证失败, 则放弃群组成员验证过程;
- k) 群组成员发布方计算 $(x'_U, y'_U) = [w]P_1 + [t]F$;
- l) 群组成员发布方计算 $e' = H_1(PK \parallel P_1 \parallel F \parallel n_I)$;
- m) 群组成员发布方计算 $v' = e' + x'_U \bmod q$;
- n) 群组成员发布方验证是否满足 $v = v'$, 如果不相等即验证失败, 则放弃群组成员验证过程;
- o) 群组成员发布方从 Z_p 中选取随机数 r , 计算 $A = [r]P_1, B = [y]A, C = [x]A + [r \cdot xy]F, D = [r \cdot y]F$;
- p) 群组成员发布方从 Z_p 中选取随机数 a' 并计算 $h = r \cdot y$;
- q) 群组成员发布方计算 $R_1 = [a']P_1, R_2 = [a']F$;
- r) 群组成员发布方计算 $e^* = H_1(P_1 \parallel B \parallel D \parallel F)$;
- s) 群组成员发布方计算 $c = e^* + x_{R_1} + x_{R_2} \bmod q$;
- t) 群组成员发布方计算 $s = (1 + h) - 1 \cdot (a' - c \cdot h) \bmod q$;
- u) 群组成员发布方发送 A, B, C, D, c, s 给辅助签名方;
- v) 辅助签名方验证 $e(A, Y) = t^-(B, P_2)$;
- w) 辅助签名方验证 $e(C, P_2) = t^-(A + D, X)$;
- x) 如果 v) 和 w) 的验证中有其中之一不成立即验证失败, 则丢弃此过程;
- y) 辅助签名方计算 $t^+ = c + s \bmod q$;

- z) 辅助签名方验证是否满足 $t^+ \neq 0$, 如果 $t^+ = 0$ 即失败, 则放弃;
- aa) 辅助签名方计算椭圆曲线上的点 $R_1^+ = (x_{R_1}^+, y_{R_1}^+) = [s]P_1 + [t^+]B$, $R_2^+ = (x_{R_2}^+, y_{R_2}^+) = [s]F + [t^+]D$;
- bb) 辅助签名方计算 $e^+ = H_1(P_1 \parallel B \parallel D \parallel F)$, $c^+ = e^+ + x_{R_1}^+ + x_{R_2}^+$;
- cc) 辅助签名方验证是否满足 $c = c^+$;
- dd) 如果不相等即验证失败, 则丢弃此过程;
- ee) 签名方的群组成员签名凭证为 (A, B, C, D) 。

8.3.3 签名过程

输入群组成员签名参数 (A, B, C, D) 、群组成员打开方公钥: $opk = (Q, V, W)$ 、随机数 $n_V \in \{0, 1\}'$, 待签名消息 $m \in \{0, 1\}^*$ 以及连接基 bsn , 其中连接基是一个任意的字符串, 并由其来实现连接能力。

注 1: 随机数 n_V 通常由验证方选取。

注 2: 另外一种处理 n_V 的方法是将 n_V 作为消息 m 的一部分。

签名过程由下列步骤完成:

- a) 主签名方拥有群组成员私钥 f 及参数 F , 关系式为 $F = [f]P_1$, 同时辅助签名方拥有 (A, B, C, D) ;
- b) 辅助签名方从 Z_p 中选取随机数 r' ;
- c) 辅助签名方计算 $A' = [r']A$, $B' = [r']B$, $C' = [r']C$, $D' = [r']D$;
- d) 如果 $bsn \neq \perp$, 计算 $J = H(bsn)$, 否则返回失败;
- e) 辅助签名方发送 J 给主签名方;
- f) 主签名方计算 $K = [f]J$;
- g) 主签名方发送 F 给辅助签名方;
- h) 辅助签名方随机选取 $(r_a, r_\beta) \leftarrow Z_p^*$;
- i) 辅助签名方计算 $C_1 = [r_a]Q$, $C_2 = [r_\beta]V$, $C_3 = [r_a + r_\beta]W + F$;
- j) 主签名方从 Z_p 中选取随机整数 a ;
- k) 主签名方计算 $R = [a]J$, $(x_R, y_R) \leftarrow R$;
- l) 主签名方计算 $e_1 = H_1(J \parallel K \parallel bsn \parallel m \parallel n_V)$;
- m) 主签名方计算 $c_1 = e_1 + x_R \bmod q$;
- n) 主签名方计算 $s_1 = (1 + f) - 1 \cdot (a' - c_1 \cdot f) \bmod q$;
- o) 主签名方发送 (c_1, s_1) 给辅助签名方;
- p) 辅助签名方输出匿名签名 $\sigma = (A', B', C', D', K, J, C_1, C_2, C_3, c_1, s_1, n_V)$ 。

8.3.4 验证过程

输入已签名消息 m' 、连接基 bsn 、随机数 $n_V \in \{0, 1\}'$ 、签名 $\sigma = (A', B', C', D', K, J, C_1, C_2, C_3, c_1, s_1, n_V)$ 和群组公钥 (X, Y) , 验证过程如下:

- a) 如果 c_1 或 $s_1 \notin [1, q-1]$, 返回失败;
- b) 验证 $J = H(bsn)$;
- c) 如果 $\forall f' \in Roguelist$ 且 $K = [f']J$, 返回失败;
- d) 验证 $\hat{t}(A', Y) = \hat{t}(B', Y)$ 和 $\hat{t}(C', P_2) = \hat{t}(A' + D', X)$, 如果均不相等, 则验证不通过;
- e) 计算 $e_2 = H_1(J \parallel K \parallel bsn \parallel m' \parallel n_V)$;
- f) 计算 $t_1 = c_1 + s_1 \bmod q$;
- g) 如果 $t_1 = 0$, 返回失败;
- h) 否则计算 $(x_R^+, y_R^+) = [s_1]J + [t_1]K$, $c_2 = e_2 + x_R^+$;

- i) 如果 $c_1 \neq c_2$, 返回失败;
- j) 可追溯验证过程(可选的)。

8.3.5 打开过程

给定一个签名 $\sigma = (A', B', C', D', K, J, C_1, C_2, C_3, c_1, s_1, n_v)$, 群组成员打开过程由群组成员打开方使用群组成员打开方密钥 (α, β) 执行下列步骤完成:

- a) 计算 $F' = C_3 - ([\alpha]C_1 + [\beta]C_2)$, 验证是否满足 $F' = F$, 其中 $F = [f]P_1$;
- b) 若是相等, 在成员列表 LIST 中搜索 F , 并输出对应的用户信息;
- c) 否则输出打开失败。

8.3.6 连接过程

给定针对已签名消息 m' 的两个签名 $\sigma = (A', B', C', D', K, J, C_1, C_2, C_3, c_1, s_1, n_v)$ 和 $\sigma' = (A', B', C', D', K', J', C'_1, C'_2, C'_3, c'_1, s'_1, n'_v)$, 连接过程如下:

- a) 验证签名 σ , 过程如下:
 - 1) 如果 c_1 或 $s_1 \notin [1, q-1]$, 返回失败;
 - 2) 验证是否满足 $J = H(bsn)$;
 - 3) 如果 $\forall f' \in \text{Roguelist}$ 且 $K = [f']J$ 成立, 返回失败;
 - 4) 验证 $\hat{t}(A, Y) = \hat{t}(B, P_2)$ 和 $\hat{t}(C, P_2) = \hat{t}(A + D, X)$, 如果两个验证中有其中之一不成立立即验证失败, 则验证不通过;
 - 5) 计算 $e_2 = H_1(J \parallel K \parallel bsn \parallel m' \parallel n_v)$;
 - 6) 计算 $t_1 = c_1 + s_1 \bmod q$;
 - 7) 如果 $t_1 = 0$, 返回失败;
 - 8) 否则计算 $(x_R^+, y_R^+) = [s_1]J + [t_1]K, c_2 = e_2 + x_R^+$;
 - 9) 如果 $c_1 \neq c_2$, 返回失败。
- b) 验证签名 σ' , 过程如下:
 - 1) 如果 c'_1 或 $s'_1 \notin [1, q-1]$, 返回失败;
 - 2) 验证 $J' = H(bsn)$;
 - 3) 如果 $\forall f'' \in \text{Roguelist}$ 且 $K' = [f'']J'$ 成立, 返回失败;
 - 4) 验证 $\hat{t}(A', Y) = \hat{t}(B', P_2)$ 和 $\hat{t}(C', P_2) = \hat{t}(A' + D', X)$, 如果两个验证中有其中之一不成立立即验证失败, 则验证不通过;
 - 5) 计算 $e'_2 = H_1(J' \parallel K' \parallel bsn \parallel m' \parallel n_v)$;
 - 6) 计算 $t'_1 = c'_1 + s'_1 \bmod q$;
 - 7) 如果 $t'_1 = 0$, 返回失败;
 - 8) 否则计算 $(x_R^{++}, y_R^{++}) = [s'_1]J' + [t'_1]K', c'_2 = e'_2 + x_R^{++}$;
 - 9) 如果 $c'_1 \neq c'_2$, 返回失败。
- c) 如果 $J = J', K = K'$, 则输出 1(已连接); 否则输出 0(未连接)。

8.3.7 撤销过程

8.3.7.1 撤销过程的细节参见参考文献[8], 在该机制中支持四个类型的撤销(私钥撤销、验证方黑名单撤销、签名撤销和证书更新), 前三个撤销同 6.4.6, 私钥撤销和签名撤销既可以是全局撤销也可以是本地撤销, 验证方黑名单撤销和证书更新是全局撤销。

8.3.7.2 证书更新撤销过程如下:

- a) 群组成员发布方从 Z_p 中选取随机数 x' ;

- b) 群组成员发布方计算 $X' = [x']P_z$;
- c) 新的群组公钥是 (X', Y) ;
- d) 新的群组成员发布密钥是 (x', y) 。

8.3.7.3 更新成员证书过程如下:

- a) 群组成员发布方计算 $\theta = x' / x \bmod p$;
- b) 每一个合法的成员拥有群成员证书 (A, B, C, D) 。
 - 1) 发布方计算 $C^- = [\theta]C$, 并发送 C^- 给成员;
 - 2) 成员更新证书 (A, B, C^-, D) 。

附 录 A
(规范性附录)
对象标识符

本附录规定了本部分中所有机制使用到 ASN.1 模块的对象标识符。

```

GBAnonymousSignatureUsingGroupPK {
    iso(1) member-body(2) cn(156) bwips(11235) GB/T 38647(38647)
    GB/T 38647.2(2) asn1-module(1) mechanisms-using-group-public-key(0)}
DEFINITIONS EXPLICIT TAGS ::= BEGIN

--EXPORTS All;--

IMPORTS

    HashFunctions
        FROM DedicatedHashFunctions {
            iso(1) standard(0) hash-functions(10118) part(3) asn1-module(1)
            dedicated-hash-functions(0) };

OID ::= OBJECT IDENTIFIER-alias

--Synonyms-

gbid-as-gpk OID ::= {
    iso(1) member-body(2) cn(156) bwips(11235) GB/T 38647(38647)
    GB/T 38647.2 (2) algorithm(0) }

--Assignments--

gbid-as-gpk-m-1 OID ::= { gbid-as-gpk mechanism1(1) }
gbid-as-gpk-m-2 OID ::= { gbid-as-gpk mechanism2(2) }
gbid-as-gpk-m-3 OID ::= { gbid-as-gpk mechanism3(3) }
gbid-as-gpk-m-4 OID ::= { gbid-as-gpk mechanism4(4) }
gbid-as-gpk-m-5 OID ::= { gbid-as-gpk mechanism5(5) }
gbid-as-gpk-m-6 OID ::= { gbid-as-gpk mechanism6(6) }
gbid-as-gpk-m-7 OID ::= { gbid-as-gpk mechanism7(7) }
gbid-as-gpk-m-8 OID ::= { gbid-as-gpk mechanism8(8) }

AnonymousSignature ::= SEQUENCE {
    algorithm ALGORITHM.&id({ASAlgorithms}),
    parameters ALGORITHM.&Type({ASAlgorithms}){@algorithm} OPTIONAL
}

```

```

ASAlgorithms ALGORITHM ::= {
    gbas-gpk-m-1 |
    gbas-gpk-m-2 |
    gbas-gpk-m-3 |
    gbas-gpk-m-4 |
    gbas-gpk-m-5 |
    gbas-gpk-m-6 |
    gbas-gpk-m-7 |
    gbas-gpk-m-8
}
gbas-gpk-m-1 ALGORITHM ::= {
    OIDgbid-as-gpk-m-1 PARMS HashFunctions
}
gbas-gpk-m-2 ALGORITHM ::= {
    OIDgbid-as-gpk-m-2 PARMS HashFunctions
}
gbas-gpk-m-3 ALGORITHM ::= {
    OIDgbid-as-gpk-m-3 PARMS HashFunctions
}
gbas-gpk-m-4 ALGORITHM ::= {
    OIDgbid-as-gpk-m-4 PARMS HashFunctions
}
gbas-gpk-m-5 ALGORITHM ::= {
    OIDgbid-as-gpk-m-5 PARMS HashFunctions
}
gbas-gpk-m-6 ALGORITHM ::= {
    OIDgbid-as-gpk-m-6 PARMS HashFunctions
}
gbas-gpk-m-7 ALGORITHM ::= {
    OIDgbid-as-gpk-m-7 PARMS HashFunctions
}
gbas-gpk-m-8 ALGORITHM ::= {
    OIDgbid-as-gpk-m-8 PARMS HashFunctions
}
--Cryptographic algorithm identification--
ALGORITHM ::= CLASS {
    &.id OBJECT IDENTIFIER UNIQUE,
    &.Type OPTIONAL
}
WITH SYNTAX { OID &.id [PARMS &.Type] }
END--CAnonymousSignatureUsingGroupPK--

```

附 录 B

(规范性附录)

密码杂凑函数

B.1 比特串和整数间的转换:BS2IP 和 I2BSP

基本单元 BS2IP 和 I2BSP 在比特串和整数间转换,定义如下:

函数 BS2IP(x)映射比特串 x 到整数值 m 如下。如果 $x = \langle x_{l-1}, \dots, x_0 \rangle$, 其中 $x_0, \dots, x_{l-1}$ 是比特, 则值 m_i 定义为 $m = 2^{l-1}x_{l-1} + 2^{l-2}x_{l-2} + \dots + 2x_1 + x_0$ 。

函数 I2BSP(m, l)输入两个非负整数 m 和 l , 如果这样一个 x 存在, 输出长度为 l 的唯一的比特串 x 使得 BS2IP(x)= m 。否则, 函数输出错误消息。

B.2 拥有更大输出长度的密码杂凑函数:HL

HL 是一个密码函数, 它基于 ISO/IEC 10118 中的一个密码杂凑函数 $H: \{0, 1\}^* \rightarrow \{0, 1\}^h$ 将一个串 m 杂凑到 $\{0, 1\}^k$, 其中 $k > h$ 。HL 是使用 MGF1 和 PKCS#1 构建的。它包括下列步骤:

- a) 如果 $k > 2^{32}h$, 输出“Fail”并终止;
- b) 设 T 是一个空的二进制串;
- c) 对于从 $0 \sim (k/h - 1)$ 的 i , 使得 $T = T \parallel H(m \parallel \text{I2BSP}(i, 32))$;
- d) 返回 T 的前 k 比特。

B.3 杂凑一个素数域的元素:HBS2PF

HBS2PF 是一个密码函数, 它将串 m 杂凑成 Z_p 中的一个元素。该密码杂凑函数的三个构建方式在本附录中给出。它们分别表示为 HBS2PF1、HBS2PF2 和 HBS2PF3。

HBS2PF1 可以是 ISO/IEC 29150 中构建的全域密码杂凑函数 FDH1。

HBS2PF2 包含下列步骤:

- a) 设 H 是 ISO/IEC 10118 中的一个密码杂凑函数, 它的输出至少为 p 的比特长度;
- b) 设 $h = \text{BS2IP}(H(m))$;
- c) 返回 $h \bmod p$ 。

HBS2PF3 包含下列步骤:

- a) 设 H 是 ISO/IEC 10118 中的一个密码杂凑函数, 它的输出至少为 p 的比特长度;
- b) 设 $plen$ 为 p 的比特长度;
- c) 设 $mlen$ 为 m 的比特长度;
- d) 设 $i = 0$;
- e) 设 $h = H(\text{I2BSP}(p, plen) \parallel \text{I2BSP}(mlen, 128) \parallel \text{I2BSP}(i, 128) \parallel m)$;
- f) 设置 z 是 h 的最左 $plen$ 比特位;
- g) 如果 $\text{BS2IP}(z) < p$, 输出 $\text{BS2IP}(z)$ 并停止进程;
- h) 由 $1 \sim i$, 如果 $i < 2^{32}$, 则回到 e), 否则返回“Fail”。

B.4 杂凑椭圆曲线的一个点:HBS2ECP

让 E 是在明确给定的素数域 F_p 上的椭圆曲线。HBS2ECP 是一个密码函数,它能将一个串 m 杂凑成 E 上的点它包括下列步骤:

- a) 设 $i=0$;
- b) 设 I2ECP 为一个能够将整数转换为 ISO/IEC 15946-1 中椭圆曲线上的点的原语;
- c) 设 $x = \text{HBS2PF}(\text{I2BSP}(i, 32) \parallel m)$;
- d) 让 $P = \text{I2ECP}(x)$ 。如果 I2ECP 成功,输出 P 并终止该进程;
- e) 由 $1 \sim i$ 。如果 $i < 2^{32}$,则回到 c),否则返回“Fail”。

附录 C
(资料性附录)

采用群组公钥的匿名签名机制的安全指南

C.1 数学假设的描述

C.1.1 总则

下列计算困难性的假设基于该部分机制的安全性,即,强 RSA 假设(参考文献[13]),DDH(Decisional Diffie-Hellman)假设(参考文献[2]),SDH(Strong Diffie-Hellman)假设(参考文献[12]),LRSW(Lysyanskaya-Rivest-Sahai-Wolf)假设(参考文献[18])和静态 DH(Static DH)假设(参考文献[8])。表 C.1 给出这些假设基于该部分机制的安全性的归类。

表 C.1 在机制中使用的数学假设

	强 RSA	DDH	SDH	LRSW	静态 DH
机制 1	✓	✓			
机制 2	✓	✓			✓
机制 3		✓	✓		
机制 4		✓		✓	✓
机制 5	✓	✓			
机制 6		✓	✓		
机制 7		✓	✓		
机制 8		✓	✓		

C.1.2 强 RSA 假设

强 RSA 假设下列问题很难解决:给定随机选取的 RSA 系数 n 和 Z_n^* 中的随机数 z ,存在 $e>1$ 和 Z_n^* 中的 y 使得 $y^e = z \pmod n$ 。

C.1.3 DDH(判定性 DH)假设

DDH 假设是假设下列问题很难解决:给定循环群 G 的阶为 p 和三个元素 $g^a, g^b, z \in G$,决定是否 $z = g^{ab}$ 。

C.1.4 SDH(强 DH)假设

SDH 假设下列问题很难解决:给定循环群 G 的生成元为 g 和阶为 p 。给定 $g, g^x, g^{x_2}, \dots, g^{x_k} \in G$,产生对 $(c, g^{1/(x+c)})$ 其中 $c \in Z_p^*$ 。

C.1.5 LRSW(Lysyanskaya-Rivest-Sahai-Wolf) 假设

给定循环群 G 的生成元为 g 和阶为 p ,设 g^x 和 g^y 给定。假设一个 oracle 是用三元 (a, a^{sy}, a^{x+stry}) 解答问题 s 。其中 a 是 G 中的随机群组元素。设该 oracle 被称为下列问题 $s_1, s_2, \dots, s_m$ 。LRSW 假设 i

产生一个四元 $(t, b, b^{ty}, b^{x+txy})$ 是困难的,其中 $t \notin \{0, s_1, s_2, \dots, s_m\}$ 和 $b \neq 1$ 。

C.1.6 静态 Diffie-Hellman(静态 DH)假设

静态 DH 假设假定静态 DH 问题很难解决。设 G 是阶为 p 的循环群。给定 $g, h \in G$ 满足 $h = g_x$, 则静态 DH 问题为在已知静态 DH 数据库的条件下计算 x , 其中, 对于任何输入 $r \in G$, oracle 输出 r_x 。

注 1: 由于这种静态 DH 假设, 机制 2 和机制 4 的安全强度可能会减弱(参考文献[5])。然而, 对主签名者的大量数据库查询, 使得对静态 DH 问题的攻击是不切实际的(参考文献[5])。一种减轻攻击的方法是在机制 2 中选择 ρ , 在机制 4 中选择 p 作为安全素数。另一种减轻攻击的方法是限制主签名者的静态 DH 数据库查询的数量;

注 2: 为了避免安全证明中的静态 DH 假设, 可以修改机制 2, 使得成员发布过程步骤 b) 中的 J_I 和签名过程步骤 b) 和 c) 中的 J 由主签名者而不是助理签名者计算。可以修改机制 4, 使得签名过程步骤 b) 中的 J 由主签名者而不是助理签名者来计算。

C.2 安全参数的推荐选取

机制 3、机制 4、机制 6、机制 7、机制 8 都是使用双线性函数。产生好的双线性椭圆曲线方法在 ISO/IEC 15946-5 中给出。对于 80 比特安全强度, 160 比特 MNT 曲线在 ISO/IEC 15946-5 的附录 C 中 C.2 或 160 比特 BN 曲线在 ISO/IEC 15946-5 的附录 C 中 C.3 中有推荐。对于 112 比特安全强度, 224 比特 BN 曲线在 ISO/IEC 15946-5 的附录 C 中 C.3 中有推荐。对于 128 比特的安全强度, 256 比特 BN 曲线在 ISO/IEC 15946-5 的附录 C 中 C.3 中有推荐。

机制 6 还是使用传统的椭圆曲线。产生伪随机椭圆曲线的方法在 ISO/IEC 15946-5 中给出, 该伪随机椭圆曲线的一个示例在 ISO/IEC 15946-5 的附录 C 中 C.1 给出。

推荐的安全参数如下:

——机制 1: 对于 112 比特的安全强度, 推荐下列参数: l_p (1 024 比特), k (160 比特), l_x (160 比特), l_e (170 比特), l_E (420 比特), l_X (410 比特), $\epsilon = 5/4$ 。

——机制 2: 对于 112 比特的安全强度, 推荐下列参数: l_n (2 048 比特), l_f (104 比特), l_e (368 比特), l'_e (120 比特), l_v (2 536 比特), l_ϕ (80 比特), l_H (160 比特), l_r (80 比特), l_s (1 024 比特), l_T (1 632 比特), l_ρ (208 比特)。

——机制 3:

- 对于 80 比特的安全强度, 选取 160 比特的双线性椭圆曲线和选取 160 比特 t 和 p ;
- 对于 112 比特的安全强度, 选取 224 比特的双线性椭圆曲线和选取 224 比特 t 和 p ;
- 对于 128 比特的安全强度, 选取 256 比特的双线性椭圆曲线和选取 256 比特 t 和 p 。

——机制 4:

- 对于 80 比特的安全强度, 选取 160 比特的双线性椭圆曲线和选取 160 比特 t 和 p ;
- 对于 112 比特的安全强度, 选取 224 比特的双线性椭圆曲线和选取 224 比特 t 和 p ;
- 对于 128 比特的安全强度, 选取 256 比特的双线性椭圆曲线和选取 256 比特 t 和 p 。

——机制 5:

对于 80 比特的安全强度, 推荐下列参数: K_n (1 024 比特), K (160 比特), K_e (160 比特), K_s (60 比特), K_e (504 比特), Ke' (60 比特)。

——机制 6:

- 对于 80 比特的安全强度, 选取 160 比特的 G_1 中的双线性椭圆曲线, 160 比特的 G_3 中的双线性椭圆曲线和选取 160 比特的 p ;
- 对于 112 比特的安全强度, 选取 224 比特 G_1 中的双线性椭圆曲线, 224 比特的 G_3 中的双

线性椭圆曲线和选取 224 比特的 p ；

- c) 对于 128 比特的安全强度,选取 256 比特 G_1 中的双线性椭圆曲线,256 比特的 G_3 中的双线性椭圆曲线和选取 256 比特的 p 。

——机制 7:

- a) 对于 80 比特的安全强度,选取 160 比特的双线性椭圆曲线和选取 60 比特的 p ；
- b) 对于 112 比特的安全强度,选取 224 比特的双线性椭圆曲线和选取 224 比特的 p ；
- c) 对于 128 比特的安全强度,选取 256 比特的双线性椭圆曲线和选取 256 比特的 p 。

——机制 8:

- a) 对于 80 比特的安全强度,选取 160 比特的双线性椭圆曲线和选取 60 比特的 p ；
- b) 对于 112 比特的安全强度,选取 224 比特的双线性椭圆曲线和选取 224 比特的 p ；
- c) 对于 128 比特的安全强度,选取 256 比特的双线性椭圆曲线和选取 256 比特的 p 。

附 录 D
(资料性附录)
撤销机制的比较

本附录给出了撤销机制的对比。

GB/T 38647.1 介绍了使用群组公钥的匿名数字签名的三种不同的撤销等级。即：

- 整个群组撤销。
- 全局撤销：某一个群组成员资格的撤销。作为结果，撤销成员不在授权代表群组去创建一个群组签名。
- 本地撤销：签名验证方撤销群组成员去创建某一类型的匿名签名。撤销之后，执行了该类撤销的成员依然能够代表群组创建其他类型的匿名签名。

GB/T 38647.1 介绍了四种类型的使用撤销列表的撤销机制，根据列表内容分类如下：

- a) 在“私钥撤销”里，撤销列表列出吊销签名方的私有签名密钥，验证方可以检查给定的签名是否由该密钥所创建。该列表可以在全局撤销中使用，也可以在本地撤销中使用。
- b) 在“成员证书吊销”里，撤销列表列出吊销签名方的群组成员证书，签名方可能需要提供证明签名方的成员证书不在列表内。根据该机制，这样的列表可以在全局撤销中使用。
- c) 在“验证方黑名单撤销”里，撤销列表包括对应于群组签名连接基的签名(或其部分签名)，并且验证方可以检查是否给定的签名是由列表中签名的签名方创建的。这样的列表可以在本地撤销中使用。
- d) 在“签名撤销”里，签名(或其部分签名)包含在撤销列表中，一个验证方可以检查给定的签名是否包含在列表中，以及由签名方提供的一项由列表中的签名方所创建证据。根据这个机制，该列表可以在全局撤销中使用也可以在本地撤销中使用。

GB/T 38647.1 规定了四个撤销机制：私钥撤销、验证方黑名单撤销、签名撤销和证书更新。证书更新是一类成员证书撤销，其中每一个签名方将更新它的证书以便证明签名方的成员证书不在继承签名产生的列表中。表 D.1 给出全局撤销和本地撤销的机制的归类。

表 D.1 撤销机制的归类

撤销归类	私钥撤销	验证方黑名单撤销	签名撤销	证书更新
全局撤销	✓		✓	✓
本地撤销	✓	✓	✓	

私钥和验证方黑名单撤销的安全性证明在参考文献[4]的全文中给出证明。签名撤销的安全性证明在参考文献[7]的全文中给出证明。这里没有给出证书更新过程单个的证明，因为这要根据具体的机制的情况。

表 D.2 给出撤销机制在具体机制中使用的归类。

表 D.2 在匿名签名机制中使用的撤销选项

机制	私钥撤销	验证方黑名单撤销	签名撤销	证书更新
机制 1	✓	✓		
机制 2	✓	✓		

表 D.2 (续)

机制	私钥撤销	验证方黑名单撤销	签名撤销	证书更新
机制 3	√	√	√	
机制 4	√	√	√	√
机制 5				√
机制 6				√
机制 7				√
机制 8				√

在对撤销机制的对比中,下列特性和功能需要考虑:

- a) 是全局的撤销或本地的撤销。
- b) 是否成员证书需要更新。
- c) 是否成员需要创建签名进行额外的计算。
- d) 是否一个验证方在撤销检查需要执行额外的计算。
- e) 是否可撤销群组成员私钥未知的成员。私钥撤销只有在群组成员私钥已经被公布的前提下执行。如果群组成员私钥不是已知的,那么该成员不能够在私钥撤销过程中被撤销。对于其他的撤销机制,没有这种限制。
- f) 如何去规定撤销签名方。
- g) 是否撤销列表具体到特定的连接基。在验证方黑名单撤销,撤销列表具体到单个的连接基。如果创建一个群组签名使用一个不同的连接基,那么它不能够使用撤销列表撤销。对于私钥撤销和签名撤销,如果一个成员被撤销,无论使用哪一个连接基,它的签名都可被撤销。

表 D.3 撤销机制的对比

特性/功能	私钥	验证方黑名单	签名	证书更新
全局/本地	全局/本地	本地	全局/本地	全局
更新成员证书	否	否	否	是
成员额外的计算	否	否	是	否
验证方额外的计算	是	是	是	否
撤销是否私钥不知道	否	是	是	是
如何撤销签名方	私钥	签名	签名	证书
RL 具体到连接基	否	是	否	不可用

对于私钥撤销,如果一个群组成员私钥泄露,它能够在该类型的撤销列表中撤销和代替。由该密钥产生的所有签名可由任何验证方连接,包含了之前已泄露密钥产生的签名。这意味着依靠保护群组成员私钥生命周期的密钥不具有连接能力。在某些情况下,当一个信任被打破,该撤销机制可能阻止用户撤销它们的密钥。

注 1: 在私钥撤销中,成员可由群组成员发布方或验证方撤销,但只有成员的群组成员私钥被揭露出来。

注 2: 在机制 1~4 中,撤销私钥的拥有者可能被“陷害”成他不是签名的创建者。如果一个恶意的实体从撤销列表中知道群组成员私钥,他可能获取到一个与该密钥相关的合法的群组成员证书。在机制 3 中,群组成员发布过程由签名方执行,在机制 4 中由发布方执行,恶意的实体可以通过重新注册该私钥获取一个合法的群组成

员证书。在机制 2 中,发布方在发布过程中执行一个撤销验证,只要发布方拥有一个最近的撤销列表可防止恶意的实体重新注册该私钥。在机制 1 中,原始的群组成员凭证可由窃取者在群组成员发布过程中获取,除非群组成员发布方和成员间有一个安全可靠的信道。

注 3: 在机制 1~4 中,群组成员发布方可以根据它所知群组成员私钥创建成员证书。因此发布方应是可信的,不能将成员置到已撤销的私钥上。

附录 E
(资料性附录)
数值实例

E.1 机制 1

安全参数:

$l_p = 1\ 024$
 $k = 160$
 $l_x = 160$
 $l_e = 170$
 $l_E = 420$
 $l_X = 410$
 $\epsilon = 5/4$

SHA-256 被用做基础的密码杂凑函数。其输出将被截断以保持最小的主要的 k 比特位。 H_r 是附录 B 中 B.2 的 HL 函数。

群组公钥:

$n =$

72E3DD15 AA189A85 D8169EE3 78E2C310 773BDA0A 8C21813F A2309CBF BA0D8BD4
64CA8774 070EE0B9 B6726805 D9D61D2D A52D0347 8447C6FF 5D297116 B5F83211
C8FBB55A 89BEE236 A4856F26 13DE5531 27F8AD43 A1716E10 009346F6 31509055
D5148DE1 5C3E107F B3F14B7F A15D4F15 4E59ED89 34B38A78 31AD0997 1871689B
7364B08E 780EAED3 DF0326A1 204ABD56 EF9853AF 172FEC1D E53253CF 08C80B2C
A25BE0B0 7AB8F661 DB1CBDC8 5E0A4E80 ED0671D3 25182293 07FDFB8B F8F77D0E
33D5CBC1 C8A8A2AF C43261DD 2351E1BB 0E62D77B 9FB388CF C18EB5D6 36E23703
503ECF2D D837D202 02678B4D C408AF9E 5580433E 61168CB3 DD877C3F 4CA4EDED

$a =$

0F530B25 19C8AA74 B8E3E632 2E7F9617 8AB8BB3C 1FAFBC4B 0BAFA43D C2D45A9A
16DB7C46 F4477DD5 C74B4FA8 37838BD4 3D6500C5 8DE91529 2EBBF49D 6A16BDA1
889B75D0 C0E0383B C8F02881 F63A9144 DDCB2E39 424CEF64 CADB4D9B B08177B0
C04DC0AD B656C4F5 2F12AC8F A17C9FCC 4A1EBDF9 859B263B 7FC817BD BB5034EB
B84BD698 FB79FD7A 45008B1E 0843F34E 39839E95 304DC1F3 0A32952E 421D4288
FC5E1001 AB16A37F 3AA6988A 13B6F6C6 D9568F3E 28A38657 45393F68 C843622B
A878922B F5E5C309 36F93406 8DBA39E8 ECD30FB1 FF4E5EDC 6EC605ED DE227E3F
4D54501B 6CA69D5F 572E2F56 AB0EDC22 A87AB7C6 7543C8D6 E465EC19 C51276D2

$a_0 =$

65451CF2 6AA403AD E92A9F13 A34152C8 041A9247 DB362D42 A60E5DBE A07E050F
518737D9 8C16092B 47ECF4AB 075622A0 FDFC20A4 C4291174 2F76DF96 1D1E840E
548A06FE 859CF811 E7AE7290 35ADA222 44C27516 27781334 6EF043FD AA9D0844
8334240A 093B8DC6 0BFAF928 4AAF9177 BBBC49BA FD4D2830 F604C8DD 6C1A184D
5087730C 99108452 3C30CD2A 3AFEB32 D72DA861 F0377CD9 5B07F52A 32ED0D28

3BF27F5C 3C3E4578 81E4BE7F 522F9522 F9670D3B C98AC2F2 291BEEEC 973A3479
D0A2D08E D4AEB51E BAAEC971 55F7E430 083955F8 A1610212 A2484FF8 74767056
27706C2A 877D1B13 25589D19 16681056 D5BE2A7B 502C2327 42873B48 07E4C1C1

$g =$

466402AF B01C8AE0 74BA7E6D 3EFF5CF1 AA754ACE F1B5FC45 C58ACCA1 4C84576A
FCC09362 5EAF074F 723F3BED 8E552E62 5C334C6B CA179580 3C018B4B 8B9329F8
1276DDDD 454E4318 6D8C4D9C 35C9DC07 70E1A1A0 EB27E5F2 17BAE6AD 1EE74712
BF21CA80 9FEA88BD 8D2F4B2A A9F50516 467AF41F 617F3DCF 8BEC54E0 B0DE32F0
D1DFFBCE 434FB1D0 A6789FC7 381704E2 82EC8859 95017AB6 BBD8D627 88AEF089
F973FC67 4192271D 3C64C750 07861325 579A63EC 60A7DBCF 23C921E5 B74CE2D3
6F824562 D405FC8C 797EF97D BB6C1E8A 7B1F57EA B0E4D063 23C5F88C ABD3A6C0
653BAFD1 155B6A69 37A62A07 9099FA11 6ACD068B EC7AE221 5010BFA0 AE861FC9

$h =$

13C4A935 45BD9A55 ABD12349 1F173DC1 75BBA355 58B5A9B4 31E22402 1EA2483D
1F9685EA 6552D0D7 8A4BA800 C12ED12F AC8458B7 B7FB2662 3C40244A 70710878
BE952AAC C3E35D93 C039EBEA 1F554473 C4904A9A A460C670 AACF7CE3 504C4916
ECFC08E2 528183D5 5DDC1AD1 EDCF645C 66ADD64B E6EF8953 F7690C41 E5643211
0982D6FE 27601EE8 8404081F 6010CB3D 034E467B F7356E48 51FCBA1B 566E3DAB
DCE9C1C8 E7276A23 04A0E066 44649BF8 CF938313 AE7F8494 F91BD3AF 2472EF3E
89B1D124 581002F0 B4124EF1 5AD5FB55 EFD78AF8 DCE73109 E4DE7504 EB458193
A828A8F1 022FED61 D70D02C6 F6C805B0 E0C71546 8D269CCD 8809A16F 4C54154C

$b =$

3FC0A1C7 C715150C 7D0DF108 DF8A0528 FE4EC7A1 48A066A3 F156C3E8 467291B3
55D47D60 00F010AB 6C341D90 C633870E 7B65BB4F 94BD7D07 9E80A8F4 55B8087B
0D82ADB8 3DCB292B 79B59B8C 812341B2 593FD3B4 1D3A7A44 983BEAFC 8C97EB02
FAF9B756 B188F365 9E65D07F 380E989D 658492BB 7AA5B7B4 141F36B8 090B0E0A
6E8FD4E2 B24947C3 13FE7307 94A4CB36 E326909D E66FD924 84DB4A71 15DDC5D0
9B1FD469 479EEE71 CF2B3313 D8208166 F2C15F4E F2DCCA79 3B7F8D27 4F5B5D85
735E04B6 A0A3551C 9BA6EE02 496C8035 D00211E1 58F86460 C07199D4 8608C4A2
58D937ED AEE23E51 7DF14162 72F5C72C 3EDCD55D 959ED4DA 05759D24 E3BE5823

群组成员签名密钥：

$A =$

375B6697 828C1CB3 6FF9E3D9 7D7EA508 D34E09AF 40ACAC31 70757709 386A6613
8D76A861 1A0C9FED 306471C0 FBD0CE41 277AE81B 42F7CA20 CB231C99 F54E4DB1
305620E7 93E8AF3A EBD922DE 45FBC66D C5DB0A70 8E7001B3 3497CEB2 D74C5652
091977B9 C4CF5356 941CDFA5 A37FF2D5 807262FA B2284DAB 3B5E68AC A927882D
E4A4882F 48F999ED C49B38F8 D42A7CF8 1955E01F 4884CB69 40D746B6 A25ECD98
D61D351F 0D2B6A5F E9D3BD81 6521D350 AA47D3AE 6CC31A9D B354D300 5E143BEE
34C8B7D2 9381681E DA5D68A2 EA63AD4A A29CE58C D9F2B690 5B42CA9F 2651DFD6
D2E0AD8F 91E3A1B0 B9ACCF25 5C47BEE0 63306943 045373E3 B4FB53E0 5149C84D

$e =$

0000000F FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
FFFFFFCEB 723BB231 7D993CAD 5EE5F0C2 4F8FD30D 5446A56F

$x =$

04000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
3E75531E 1AC338DD 9B150FBB F9B2F984 40A30BDF

签名过程:

$bsn =$

00313233 34353637 38393130

消息 $m = \text{“abcdefg”}$

$f =$

453FD0DD FDBA8F31 2A1CE338 F2767D9A BCD031FF C42C45B2 B0328B9D E410BC1F
C23D3FF8 AC3807C6 52ABACAC A579005F ADF45BF4 3D208C0E CB717317 4F4D2AC6
F9313E2E 41EE1F3D AF3264E4 89848B11 A5570E97 5BEFD851 5127FD0D CA78FB62
7000EF26 64AD94C1 802D0ABB 3CF81E74 FCAF9930 DB1DB12F 3805F1D4 A540F706
34DC2953 0D09DA60 D3337D20 61F6388C D0D49087 2B9C9D0E 2F2CFD04 BD167D28
F84E0BDA F3227C69 52586FF4 FF2418C1 67B067E5 55D4A77E FB3ECA4E 58384605
5C8FEDC9 BD3F3E4F DFFE6F95 FF4E8199 73D54ECB AE183533 6B4BDFEF F9F3FA2F
F677450F 7247D2E9 EA74200F 325492FB 14BF85A0 7C249E54 7635BBDC E9ABDF77

$w_1 =$

903FF56B C8865FCB 058A162F F71734F5 6AA67822 CFFDFBBF 266D6EAD 3B81155A
4E317776 D9B2B351 59CD0C9E D8A3766B 9781EF3C 8CE19EBD 0DD2C5DD 45D5816D
6D5085A2 060CC78E 1D098071 7DF8D934 D439E460 1A360FFE 88C3841B C4D98091
CAB4726B 8691FCF6 FAABC40F 5938A9FE 01E7A93B A77AFEAA 29BB2D9E F7BCF534
BD895888 E1DEE337 9A647719 7D70944C 7F3A13E3 44F88E81 639DAC44 128439FA
79C05588 45375525 00C1D51D 56143961 740E394F 61C2DCED CE468A85 CB869825
95B80F52 F966A15B 4DEA4D12 359B8FAB D9277AD6 561974B7 2BB8D7E6 2785DC5F
FF502DEF 7B1630BD 4A430084 CE13C685 0432A943 DDB1B767 2F1EA8E3 4E74B149

$w_2 =$

3C596556 AA7E429B 1703FE21 BEE7D260 3092AD99 12FD90EA 6671B8EA A901BBA1
C5F2E3A0 0EA6CDAC 7B5A9E0E E3748AC3 09F69D51 D8B7F423 2D5237EA 34D6FC21
9EE94387 1370974B 9B6C8E49 1E9A0C92 FF4D1C51 759891F0 1E5B9A5A 04AA5B64
4A228412 056C1D05 95F06020 8570587F 96D49E98 665CA6D3 866F8249 2FA2E1E9
0C87A891 6E15BF72 927C4446 E53E0F5A 09FD6D16 158B9016 0A726C13 9214A0D4
DF2FEAF6 AE223430 D97424C0 20209D49 131169D5 10EC35E6 41AEF4B7 B2B0CE8F
A8C73910 AD220D0A 168CD86F 8EA09187 12121EC6 4FA286AB E9DBA4C1 B1215D58
6E5F4E40 F9FFDAC3 F51AFBBE B1320B28 824F53B0 DE79FE63 9CD66590 CD56F2A6

$w_3 =$

2763853E 200092FB 842CC54D A92B377B 4E2FD10F DEE0FB24 E5718D59 7AD0EF99
8023E2D1 424F817B 1ECF7017 C470878D 8FB5F2E8 694D11F6 631F075E 5B2617DA
99A8BCD6 BD6281D3 E86C58DC B4031BD8 A5756F4C 20794583 C1F61D83 FD81E025
7F160843 B38B4DD5 CE8423C2 3DBDD7BB C2CF698A 78D72132 442D3A83 D963D3DD
87538F97 3EBE1D0B B147C9F3 FAC8EE84 8D25EA64 004F1254 D325844B CBBE8424
06FE0120 A5E14703 9B246776 04875F24 9433CBE8 612A06D4 B817B0EA 66C8F3F7
80A35798 0425CC2A 8CE6A6A4 24BCFEF3 8F7F0697 E7A2ED23 6D59EB3D FFFCB69B
92EF74D2 33909BB1 31709001 1AE931AE 6E8C1A18 D96AAE32 BDF504CB E247570C

$T_1 =$

5C96F1C2 EDA116B5 6C5A28C3 8098950B DE084521 6E1A5FA2 0F985999 064FCCC1
 9946B54C 9A21F41B D5713079 16C8BF52 10FFD01B C3C0BA65 09659C87 68B17137
 22845000 4843DCDE 201F9C01 C1A48948 A165A2E4 13CBC416 F657721B 4AB69F3D
 92EC3039 E0E82620 03F69444 275C36FD 20A8DDB7 4BE29929 66A2DD7A 8663837E
 05DD224A 8DA92BF4 1CD5F9F5 31DF5DCB 3112F0C4 229D7DDF B0FA13E4 5CB5B794
 DCB40C2A 3E42D593 E13AD04F 8857D361 58BCE8E2 64C07A18 0C81AD49 B8EA3790
 DE52B00F 5B403AA3 26307A79 29AEB630 53183B2F 59A6AB34 B45CB1C7 AE76F402
 FE9D7396 FA3A140F ABCF31A4 A6D1BE6C 55B89D4A 6B3A4DC1 51242787 56E7A4AF

 $T_2 =$

44F0115C EFF930AC 803726E6 614B46F9 20310BD1 68956D60 3E771237 7B512D02
 CF04A986 943BB3F8 5F4C4D12 4438AF50 A1AD13C6 D7FF5666 74F2241F 6386CA60
 55DF0BFB F5860275 C450CA14 FAD2D40B D57245F5 B029A009 AD7CFF15 66C57517
 15591D64 93473277 9EA974C2 E21B3467 E42AA151 5BE575B6 686A666F B9ADD370
 316007B3 312887E3 7BDDA993 049A09C0 C884AA55 FE752E20 BFB6CE7A 9F12F5A2
 3D9053EC A7EB8823 21BC9B19 BFE59B4C D3BD32C0 C12A52A0 FD3C6B02 1386A207
 1286E1AB 4614889C 941EE9D5 13A7B3C7 32B0718B 4BBFE878 A2793DAD C561A1B0
 3AA80A55 80004C58 1D390C69 2577875A A0324F75 05A89AE4 7AB61455 1F0B0550

 $T_3 =$

1910180E F5ED5B8B 166E43F2 CABBBBA9 038FCFA2 DBE9556A 54E8D5F8 523577DD
 5AC89BBE 510809A3 B4B8A6B8 F8AC3A0E 9AE6B964 5EA1B41C 5E2577A7 0D080FE9
 675DAED6 65796AC6 46CBC74B B4B2D427 A805C096 E4DEF4FB 707B4EC7 E3965EBE
 7907F09F DEF70AD6 1CF8CBD4 39B3AD69 76EFDB7A E36C6CC0 1BACD171 090392C7
 86453340 3A4AB456 A2E16DD9 B84A6C41 24870E87 DDC8578E 72E055CF DE15F7FB
 C4A7000B C9CF666F 9B5877E4 8117708E 8B5071FB A2BBFE4F ED8D346D 25A19FBE
 609D067C E946CD56 C6B4A81B 4A8089EA 90904077 4792631C B868E450 70427A48
 16B6832E ACF5C5FF 864C4F3E 6B6F78D7 3683E379 B1186EAE C9C61F2D 7E6FB0C7

 $T_4 =$

2CD4FD7D 0B491E59 C73B5083 B64BA089 176B5C46 A20FB06D E1B104B6 1F248CB1
 10F7C259 8C153A0C 8C9D5281 480B10CA 0C587174 573F70B2 C54B3C91 28CB0AE8
 4155A699 D29C5028 796744C3 685AD22E 4425E2E8 9789DB76 FA1C0119 12817997
 2460EDF1 0DC6B484 5F3CD951 A8348BEB 9C88FA17 579A710A D66C7E90 BA84B724
 A3645C41 5DAC6939 0BB73E83 9B9ADFC3 F6118304 3BD249AF 4A14C058 8EE8B0D7
 59487B11 D35EB9F0 DFEC0636 925EF3B3 BFE3CE98 72CF73CD DE2EEC2A A251B984
 E206043F 9061F375 25491E18 3295C134 5500997D 7D74F92E ABA5915B BA993D12
 6F9756F4 C8874899 F58D6F25 E28640A9 C3A8AE11 F1E56338 62C62308 4ADDD1FD

 $r_1 =$

0897F8E4 9394A6E1 BC625487 A21187A2 DA10383A 2AC272BC D80CDD35 A346363C
 672CEF42 7A69C199 9E2F998E E40C3D39 3A250520

 $r_2 =$

000080FC 06D07109 8A6F7870 6F973875 EB851D99 15C3B5BB 656DFA7F 395066DB
 8AE2223A B1D2E6C2 86778413 78F2F662 E3A694D6

$r_3 =$

00000069	63C9FDDDB	E0BD59A0	6BE8CE73	3BEE8F55	EEAC8C12	63586346	F5628510
89C4C6D1	B323410C	350D8F1D	65433BC8	F5090639	9C57EE0F	F5537A7C	510F1BED
C854DFE2	91970C9B	F40A7D3A	0465BAE5	0A290176	1B935C5A	D88641E6	DFDFF3DA
929BD6A9	AF1D7E28	D04D6676	32695AF6	EFE4F0CD	29C9394A	A5594E0F	6C5292F7
D3085E44	AD9FDCD1	4F0FCB5A	F7C15B1F	F917E39A	B685E634	6B3C8A58	2C65AF39
9201F20F	C5286DB0	5CE5E069	BAD92CBE	E80CFC35	8B0716AA	93334E71	C622EA15
DE2D4E90	39B4F2E6	52D2E942	7CD5FE33	B2139AA3	0A4FA342	4F575CB7	08F32014
E42083B0	C5FF3E7E	A1FE46C7	3E22C48F	B1AE2472	E7D5A375	4E6873A3	098FBA71
D24F7E5B	BAFCDC9B	5D092619	326BDB7B	7B9D9EDD	3FA7D6BC	F4BA414F	BF3AB03C
E1D9C5B1	4BB6E2EE	9A39EFDB	9CB67682	EDD9574B	452F5C7F	D40FD5AD	329C980B
7B4BF5AC	FCB933C1	E5C5CF7C	EE325304	DCE83663	563BE323	CC8BD21C	

 $r_4 =$

0000003C	8FB37C34	73F651E0	1300DB3F	61BC5162	70FC2E62	417B13EE	07AB3813
4F458F4D	C6B5163A	B1FDEF00	8252A446	6B2E2D25	3678F22F	B9034B1C	158AB04C
F47A72EA	4ED5F469	77BBC48E	1EBB6906	00BE40DB	460F0B27	A6A5FBCB	4648B7E1
8C9A03A7	F329EF53	9070AA9F	74CB2735	07A0A660	42C5A328	E829B282	31768FEF
7085973B	580F68B9	52B4B02B	8F8A894A	50277382	0E140885	5DCD5C53	84E401AD
C471E646	D02A1AF7	9366C86D	50655EE1	211AF90A	4FA2454A	6FB49379	0519E780
E9607154	33743550	43285DD8	E37F247F	E3B4DF11	573E4C91	2B1677A0	14E71A45
0A8690EB	D1836D9A	7DE9A132	26B9E50F	562F64E5	46233DF2	7CE4AD46	7A7884F0
E7641617	00B38DC3	A1A714CA	B99EF98A	8D9D330F	37A81605	F33A5942	6BF4D16B
67734E1A	C3C2F7C1	41DD9982	8BEFFC70	977A026D	9FAF50BB	2AFD9BC1	227E652C
6D9AC4DF	3C1A4DBC	8B465C82	205653DE	AF2F1408	B620BE84	29BB887B	

 $r_5 =$

0000003C	9621E1F7	CA5DE96C	DCA4AEA4	C1737836	31E69FA4	B043B89E	AF00749D
637AEC07	82D38F1E	AA92033A	04D754E3	2501B00F	94DE953B	A920F18D	EBCE7B35
F916A8C7	05FA2A06	C0E1F8B4	113AFF82	082B7A8C	167C7CF8	7CC7A6DF	AC908E2C
7C39AE54	8E3DF467	57C97E60	6443AA60	C286BFBD	ADA8BDC7	658929D7	83328942
ADEDB7C0	4D85E20D	72066918	FF01D3E0	4994C0A5	387DCDAC	7AEA530A	6999C798
8AD99B43	7EB97788	8E5DA30C	BA911F81	30216D42	E294C33A	AB097285	7AE86A06
4A3A42D3	D9839BAC	F473617F	BEDFF35E	14546C4E	4098EC16	5258A309	585DDC7A
632E254C	6C2CC0E0	C3760A10	01BA6A9D	D9DAE9EA	74BE20EC	49AB2568	B4DBC913
6A8E87BE	BB286EDA	595BF8D9	2C386565	172FDBBA	492EDECA	01C08C16	2140D08B
F0D2235F	C566365D	E643A9D7	9ADFBE93	B0BCDCC0	99B48F66	4398E5FB	

 $r_9 =$

08EE4D7F	CB83E628	8FFFE433	7A4E6501	183249CC	B3B5652F	F5F80E11	EE67A097
60BB53F7	836C86E4	99FC68F8	D7CA04B8	8507BCF1	9402591E	6F39177C	8B2BA0A9
64A22088	25847FE7	644C5E3B	DAD186B9	E9769DAA	F4FFA74F	DDB2D3EE	30913A7F
5F194D3C	38E62D4E	C47D0D15	37BD1CE5	A51EB01A	6F7F8095	196B9E2C	84E73422
2DC41C5F	CDBE2ECF	144BAB4D	98A29E08	AD551BD7	E3B6BC53	896DFA08	AC56D11D
17C88766	5E51D73D	63C78E17	4038927B	1B54138E	FAC0EC23	1DFCF6CB	DB8999EE
85F7EA55	384543E8	FABDE89F	1A28F6B2	FAF07BE3	690FB798	BB84EAEE	D86E3909

7D4506D7	1C3ECAD4	BFFF4A7E	048A7AFA	D874A576	DA3D87F1	DD853375	504EC9DD
326C148B	E2383202	58CE6988	2357E719	37F77D1F	6B565D11	F272F222	1864CB97
1A10F637	A10BEE84	3A945447	42675D1B	8482B282	B2610E4C	194AC612	A324E539
76BD228F	EA178985	5098BD20	496BD782	E937FC79	C51617E9	A175E20E	049ECAFO
9F6318B4	89E9F71C	51FCF1B6	0EDAC6F2	091BF7AA7			

 $r_{10} =$

00EE6A20	56206A63	78E71D92	A4D96CAB	C820D0BD	0B43A5A3	9E9CB7C9	681B92E5
BC4187C4	8A7092E3	20F0D654	DCF49D57	08A55292	A2BD27BA	A7FAC208	234CB3D0
F6436D72	2DE31ACB	66410105	D688454C	6D2BDD33	AB3C98BC	E53363E8	F52EC55C
E3ED40EB	FB145B7C	63F7C5DD	74293B51	4DB2010A	55EF1AE3	FA319866	ED327FF8
E0A9A05A	07B446B1	50823FA4	B26F9DE4	1F1A4FEA	035B0675	FD7408FC	129106A5
2B81DC03	5A844549	0071D608	81F99946	FB864E56	FFBBE04C	216E9B2E	7ADD5791
A40751CF	0E6D8FBE	E296FCAD	5D7DEBD6	F3CCC79A	4DC93242	A540583B	4CFDD854
565061C5	B4DB9FDD	B336F735	05AAD762	B8230813	D09E21C7	20561690	1C3F4D6E
C89EF8CB	D0E22EBF	28E00511	3B95172D	9E66BD60	97545BB8	B04A1442	D25FF5D5
7AAC0350	4980AA5A	41F62DD1	7EE4B3A0	69F04861	EA51D8B0	9D83DC8A	53C251AC
FE7D30B8	6D98D3E2	53A7AB48	F4FDD5CE	044F541E	4C774873	4FFEDDBD	13E94D98
9B9CE959	A27CCAF0	2692AC8E	A80C59CA	D6FCDBD2			

 $d_1 =$

047E2731	97415CF2	5F43729B	0DC37A27	24EDBBA4	335B4ABF	A56F5F6E	9E056590
139E0310	662D903A	939DDCE9	5D24FB44	B2AABAAB	47D2A2DB	1299AC33	49BA1FDD
31C7C571	5EB626FB	0E127168	355B1CE0	2964DC5D	B86D27E9	70BB21DC	815BECAF
493BC724	EEA46539	C82CFEA5	71FAFE42	B0A64782	0600AFD0	5FE5ED0C	550545A7
213BB3A1	F08BF173	FB533BBC	4E28C23D	313F4EC7	0E636B09	C190546D	37D6126E
D4764E97	7E7510BC	0D9D3A0B	12C36B58	5D1C3786	001DCDDb	08A4F5BF	B048CBFD
36D675B5	66AB4903	9A74D0BD	8049C5D8	A00B4254	22C80554	91FEFA3C	6BD9F690
ACF4D989	12B66E19	970A4A69	BB018268	452107F6	E2C5DBB2	CD3E079D	CB49DAB8

 $d_2 =$

2E832286	06A065EF	321A4544	B83219AC	DAEBC746	CC4EE586	86014626	329CC81F
945616AD	7D7D4C09	712EB9AC	DD63C34D	832B1BF1	2663BED2	3507F7E8	5F3F3E62
558064F5	C4230B33	F2774F2E	79BDD0AF	A35944D1	964C5322	14F97B97	D00D8554
D41133F7	44C4CA80	B24562F2	73EC44B4	3A4884B3	6D04CCEB	1E43D619	1228684C
28483E08	5F488AA9	1971576C	26867706	53853295	403AF145	34BA1113	2DDEA273
B4E12435	9670753F	688A7921	32260700	97D98440	38A3F6C3	C20D0C1D	49C9BAD9
EAA52AB3	216DD002	DB8FFB88	7547D0CB	C8D1DEE9	B5F4AB17	85F12519	E30ED49F
68E0EE44	CF595DE2	3904D788	F9422497	1B98D3D7	38E7AFD0	336A368A	F7CAD447

 $d_3 =$

5ADD5BB3	018836B6	37F20D00	EA396C0D	6AB0F07F	01026D36	BCA1BDB2	CC600C71
B1565B13	1537A8C5	3DA7998B	99234628	77BF3D27	8FA21AFC	BD3D9177	4DCA8485
6C35547B	C021A0F4	571EB589	4FDC9B0C	71EA4844	C5202753	38208D2B	DB5AF904
F47C3E69	98003EC4	37D84037	ED624AC4	4487781C	9E6362A0	23ABA6ED	1A6FCBC4
5FA6AA66	B58A51F2	61FFC110	72532411	6A5F61C9	88C5072A	71BE1361	A91A1368
1E7BF74D	C6178691	88385F53	91D6E02C	71D58C28	9746ED9A	1D6A62D8	529B7518

85E13652 C1AC61CE 7E2C09D0 42EAD4D7 5035FAF9 D2A10865 1074F3FE 169E228A
 35EA55A0 676D8DBA 8CABB721 16C1E6C9 2360E361 E674DC1C 000E0A4D 6DBB1E24

$d_4 =$

41C114FA FF5C1358 61E6DB82 32C9D368 5B1FF404 C9273A68 16A206FE 77E33E4B
 E83A740C 48D9FBE7 AE50FB52 661BB60A 67D8E512 0F500CE0 FCB98A0A F3A05B99
 24BCBB4E E655E426 3827FB1B 98C51D92 ABD961D1 D117F248 B0E078CA 7E40CCA9
 E4BB0EBE E5E197E3 97B62D1B 4778B2A3 7E4973D7 B17C91FE 68054367 77637376
 764E9336 7B451F7F 01C67108 77380C06 3701AA59 58144F2A B604FFE3 1B450C3A
 C2C1DF92 B101B1FD 21F78E15 AE7F7C4A 33B38157 ACD3D2EE 31C4714F 0B79F594
 862B5C9A 11BD89F8 94ACD8DB BB310CFF 203CF0E5 41F0E2D6 138E0B6B EBE627B2
 471098DA 388F71C1 46FD4B03 5EDCED94 7005BFEA 79EB30EB 4313E82C F6877C83

$d_5 =$

4170D919 99B8BD2C 28CFC6C9 6682C0D4 570A2DE3 46734216 10F5F5E0 B8FE0C73
 55B1A758 4162D5BF 3CA97EF1 C9E58D0F 59C2AC71 D293DC4B B79B80F2 53322535
 8C93D00F 152004A7 9DA6B065 9096A5A7 12650702 E5571C2B 0A106CF8 7271B0D6
 ACBCECCE 4463ADBC 1FD2A484 E91C4AB6 03F1507A 6B820C25 84CB3AE7 765194F2
 2C1EDD8D E3A0BA35 449CD004 A5EC30EE F45230EB CECFAFA1 1FB8E637 FB9C2C75
 C0B24140 C2A6CE20 BF9B569D 583C1C01 1F821EFA 16D3B792 FD33F388 E66BAC54
 92FFA95D B2AC0CA2 5582FFBF 565C8F1C A9499107 59A6E101 9894BB57 C1A5A030
 6CEE6326 EB35193E B245A3AE 829F6929 FB5EF611 0B84C4A6 03316161 3F998558

$c =$

44F1BC8E 392071C2 AECCF97D 770758E9 15941091

$s_1 =$

0897F8E4 9394A6E1 BC62555C 0057F04D B7A3EF48 04794086 3C2F8F79 6BEF2016
 C2015954 B6A05135 682FBBEE 9E929EF7 549E6141

$s_2 =$

000080FC 06D07109 8A6F7870 5EC513ED D103D06E 18EEFE5D D6476614 439C5233
 E76061BD 43FD44D8 9634FF9B E4A52A3C EEA2EB87

$s_3 =$

00000069 63C9FDDDB E0BD59A0 6BE8CE73 3BEE8F55 EEAC8C12 63586346 F5628510
 89C4C6D1 B323410C 350D8F1D 65433BC8 F5090639 9C57EE0F F5537A7C 510F1BED
 C854DFE2 91970C9B CD314994 3CC47429 6A435E36 CFFABD17 05B76F66 F30C7312
 308D384A B9BF2F70 0CB36A28 003BD1E7 FDE5E58F 255B8AC9 5BFE15B8 C846CB15
 4B1CB4C6 1C4264E0 AA637536 9C168394 B333CCB9 8A14F16C EB6ADD74 5F7460A6
 92F73E49 A48F4BB7 1FBC8FFB C0ED23DA 05BA14BC 5F860AB7 191536F5 6A73B95F
 3C76E954 2012C4E5 07FA95B5 572A7A16 0B1CC6A5 4395C6BC 4AB4AF75 74796CCE
 917D29DF C7AD4277 6151638F B222F7F3 914E6DC6 E19C4F0D 742F6E7D 53A7E2EA
 E7B6AA16 73018392 B8F2D1A7 03272A5C B04DB84B 97742D73 6C298CFD 62CB1CDD
 9412668A 27E262C4 B3126854 2AA44B3D D163E592 1EA3BDFC 905B017C OCA856DF
 7B1A8828 3735F33D FDFE0DBE D478915F 49085152 2551A40A 962AD7C3

$s_4 =$

0000003C 8FB37C34 73F651E0 1300DB3F 61BC5162 70FC2E62 417B13EE 07AB3813
 4F458F4D C6B5163A B1FDEF00 8252A446 6B2E2D25 3678F22F B9034B1C 158AB04C

F47A72EA	4ED5F469	677B0907	8561F5A2	7E332130	63EC8ECE	269BF616	4A52671B
6A2445E2	EE0E714A	5FA0B8DE	B708A11E	E4AA1B99	944485E3	C6A7EA17	30CCB765
AA0C0695	CC0E1FC1	5884679B	A55F4689	90CE5CB0	9CE361A2	509FA781	8B987E5C
0E7D6177	2AC8349B	4FA5DA7C	92B7BCB4	9E88CB6F	0DD69CCE	51616DF6	0EED1772
5D36C1FD	D2DF3D20	60E55CAF	6391ED4E	6A3E9432	B814C504	6A5D0F03	ACB76B26
BAA1A765	9C9AE81E	9092BE0C	2B677B5F	C54795E4	3B7C82B6	8CA85D66	54E32B05
D2EA8898	433529CD	A7887C9F	9E232773	FF516588	4287F888	1BDBC333	2E061F3A
DF0338F1	9AADD03C	61006A01	F3ADBB1B	85F9318B	0DFDD509	A46D4140	4BC3B6EB
3C2DFB4A	497775A4	7D19B03D	0863334D	408650ED	4DB36968	8659B875	

 $s_5 =$

0000003C	9621E1F7	CA5DE96C	DCA4AEA4	C1737836	31E69FA4	B043B89E	AF00749D
B6FEF425	ED3CB308	E5888F5D	D466CCC3	82A7BB7D	8888ECEB	939324CF	D93B1AB2
637AEC07	82D38F1E	9FF66222	1D035EBB	6BD76CAA	A125B4DE	5D62DCDC	BB4EC542
427DD4DD	E1DD1DC0	FFF67DB7	833FAA46	136AAF5F	CEB08AC7	B66CD78A	126F21E6
3CFF6E71	0873B45E	169E548A	ADC20832	5E90DCB7	8145A73B	F2CBD66A	A4B94186
0C2048D2	3EA47913	998A8BA3	64656CAC	B5D2CE9F	548EBCF7	7D611286	265892E0
CB47DCA1	ED2B0B91	60A331CF	EB94C1BC	EBD37736	98B59381	DAB2B403	ECFAA9BE
CEB9836D	820A1F93	FDDC31E3	9D02FA1B	D0E74BAD	66111F67	91087BC4	2B8C23A2
F05BB100	9D5D66D6	D2F288E5	5A59C676	122458C7	3206EEAD	9FDE6924	17ED6869
11EA294B	6354D7DA	16D4CC4A	46D94E2E	131BA09E	166CBD72	EB227986	0351D0D7
1456C4B9	7619D2E2	A5682C5C	36772AC9	96082178	C4031478	54CFD82F	

 $s_9 =$

08EE4D7F	CB83E628	8FFFE433	7A4E6501	183249CC	B3B5652F	F5F80E11	EE67A097
60BB53F7	836C86E4	99FC68F6	6A36CA5C	0AF35137	95A82529	B5AF233F	5E3E78AA
976A1402	049A99F8	0E677327	4B4827AD	C806C28E	68788343	B70D7791	C6FAE726
AB02551D	2AEC4D5D	9428B328	642BAE70	FCB083A6	C38F3C22	BBC7A2B9	5A6F4DBF
87C70B3D	2695A262	033092D9	491BE4C5	CC3CEA05	2FDDDA73	4FCC87EA	0463B8BA
7E16CA28	8CC6C87F	B4548FF6	0D2E62D8	25C5454C	C081295E	65F96813	38282808
3588F504	9E1BAE7D	7B9B018B	F3DFDE8E	A885E251	113327E4	0A8306E3	095208A6
7C5054FF	F006C18D	F08E891E	DF1CEF73	2406A8C9	541D9E2F	7FF48214	992F7F73
C854900E	73BEBEB0	E53B4F52	84DB4105	A8BD4619	3A3F980E	906894B8	5684781A
81376C58	E85EA4CD	A529C378	A6FD3ACC	94E186F0	DE184555	587E3E74	A9554511
7FD2062E	299652A9	49E8589D	77EF42D0	F170D665	19669936	04C617AC	C854023A
03B3E6B5	234D3935	49F1DD00	57E668C8	4B5E1110			

 $s_{10} =$

00EE6A20	56206A63	78E71D92	A4D96CAB	C820D0BD	0B43A5A3	9E9CB7C9	681B92E5
BC4187C4	8A7092E3	20F0D653	D8E8E4ED	730E1C5A	7A0B2D0C	85D2FC70	22AC5881
36DE6110	06873E7B	148920A4	D9CA8159	E2A41F04	64263D41	7C5AC822	33A5583E
C79E5376	EB592D7B	730E4517	351162FB	BC6C6562	028A41F6	E404B49A	41D98969
743C7F68	8B7934C8	2973D044	7AF3649C	F7A4F386	7B548A55	E427D3EA	D64082E7
50C9DE64	D09FF093	098A4483	58F240C5	D56CB8CC	42ECA051	82D11A56	5D388DA3
1FF61E39	4FCF402C	CFC23B3A	C81B8567	01FFD8F2	0D1EFEE6	DAC56B7F	3DAABF4D
7F0415E7	C2EFA593	108FEC47	9D7D1070	DAC9A75F	32E7BF40	D5983FA7	CE853612

16432BE0 6EEFF996 9F57205E A887BAB1 14C35619 90E4A91A 46FFE07A 6E409448
 5530B621 2E4EE66F 7D3D6C43 E3943C00 A0441F48 86C5CFAE 63E5D5E6 873AE2ED
 5DD79A03 1A997D56 33DB67B1 0A4EFB65 7695B1AA D34CFDE7 83B5ABF1 8522D526
 185872D8 4149468A C4D12A19 99D85B32 1ADBCB38

签名: $(c, s_1, s_2, s_3, s_4, s_5, s_9, s_{10}, T_1, T_2, T_3, T_4)$

签名验证:

$t_1 =$

047E2731 97415CF2 5F43729B 0DC37A27 24EDBBA4 335B4ABF A56F5F6E 9E056590
 139E0310 662D903A 939DDCE9 5D24FB44 B2AABAAB 47D2A2DB 1299AC33 49BA1FDD
 31C7C571 5EB626FB 0E127168 355B1CE0 2964DC5D B86D27E9 70BB21DC 815BECFA
 493BC724 EEA46539 C82CFEA5 71FAFE42 B0A64782 0600AFD0 5FE5ED0C 550545A7
 213BB3A1 F08BF173 FB533BBC 4E28C23D 313F4EC7 0E636B09 C190546D 37D6126E
 D4764E97 7E7510BC 0D9D3A0B 12C36B58 5D1C3786 001DCDD8 08A4F5BF B048CBFD
 36D675B5 66AB4903 9A74D0BD 8049C5D8 A00B4254 22C80554 91FEFA3C 6BD9F690
 ACF4D989 12B66E19 970A4A69 BB018268 452107F6 E2C5DBB2 CD3E079D CB49DAB8

$t_2 =$

2E832286 06A065EF 321A4544 B83219AC DAEB746 CC4EE586 86014626 329CC81F
 945616AD 7D7D4C09 712EB9AC DD63C34D 832B1BF1 2663BED2 3507F7E8 5F3F3E62
 558064F5 C4230B33 F2774F2E 79BDD0AF A35944D1 964C5322 14F97B97 D00D8554
 D41133F7 44C4CA80 B24562F2 73EC44B4 3A4884B3 6D04CCEB 1E43D619 1228684C
 28483E08 5F488AA9 1971576C 26867706 53853295 403AF145 34BA1113 2DDEA273
 B4E12435 9670753F 688A7921 32260700 97D98440 38A3F6C3 C20D0C1D 49C9BAD9
 EAA52AB3 216DD002 DB8FFB88 7547D0CB C8D1DEE9 B5F4AB17 85F12519 E30ED49F
 68E0EE44 CF595DE2 3904D788 F9422497 1B98D3D7 38E7AFD0 336A368A F7CAD447

$t_3 =$

5ADD5BB3 018836B6 37F20D00 EA396C0D 6AB0F07F 01026D36 BCA1BDB2 CC600C71
 B1565B13 1537A8C5 3DA7998B 99234628 77BF3D27 8FA21AFC BD3D9177 4DCA8485
 6C35547B C021A0F4 571EB589 4FDC9B0C 71EA4844 C5202753 38208D2B DB5AF904
 F47C3E69 98003EC4 37D84037 ED624AC4 4487781C 9E6362A0 23ABA6ED 1A6FCBC4
 5FA6AA66 B58A51F2 61FFC110 72532411 6A5F61C9 88C5072A 71BE1361 A91A1368
 1E7BF74D C6178691 88385F53 91D6E02C 71D58C28 9746ED9A 1D6A62D8 529B7518
 85E13652 C1AC61CE 7E2C09D0 42EAD4D7 5035FAF9 D2A10865 1074F3FE 169E228A
 35EA55A0 676D8DBA 8CABB721 16C1E6C9 2360E361 E674DC1C 000E0A4D 6DBB1E24

$t_4 =$

41C114FA FF5C1358 61E6DB82 32C9D368 5B1FF404 C9273A68 16A206FE 77E33E4B
 E83A740C 48D9FBE7 AE50FB52 661BB60A 67D8E512 0F500CE0 FCB98A0A F3A05B99
 24BCBB4E E655E426 3827FB1B 98C51D92 ABD961D1 D117F248 B0E078CA 7E40CCA9
 E4BB0EBE E5E197E3 97B62D1B 4778B2A3 7E4973D7 B17C91FE 68054367 77637376
 764E9336 7B451F7F 01C67108 77380C06 3701AA59 58144F2A B604FFE3 1B450C3A
 C2C1DF92 B101B1FD 21F78E15 AE7F7C4A 33B38157 ACD3D2EE 31C4714F 0B79F594
 862B5C9A 11BD89F8 94ACD8DB BB310CFF 203CF0E5 41F0E2D6 138E0B6B EBE627B2
 471098DA 388F71C1 46FD4B03 5EDCED94 7005BFEA 79EB30EB 4313E82C F6877C83

$t_5 =$

```
4170D919 99B8BD2C 28CFC6C9 6682C0D4 570A2DE3 46734216 10F5F5E0 B8FE0C73
55B1A758 4162D5BF 3CA97EF1 C9E58D0F 59C2AC71 D293DC4B B79B80F2 53322535
8C93D00F 152004A7 9DA6B065 9096A5A7 12650702 E5571C2B 0A106CF8 7271B0D6
ACBCECCE 4463ADBC 1FD2A484 E91C4AB6 03F1507A 6B820C25 84CB3AE7 765194F2
2C1EDD8D E3A0BA35 449CD004 A5EC30EE F45230EB CECFAFA1 1FB8E637 FB9C2C75
C0B24140 C2A6CE20 BF9B569D 583C1C01 1F821EFA 16D3B792 FD33F388 E66BAC54
92FFA95D B2AC0CA2 5582FFBF 565C8F1C A9499107 59A6E101 9894BB57 C1A5A030
6CEE6326 EB35193E B245A3AE 829F6929 FB5EF611 0B84C4A6 03316161 3F998558
```

 $c' =$

```
44F1BC8E 392071C2 AECCF97D 770758E9 15941091
```

注：该数值实例仅供实现者验证机制实现的正确性时参考。

E.2 机制 2

安全参数：

 $l_n = 2\ 048$
 $l_f = 104$
 $l_e = 368$
 $l'_e = 120$
 $l_v = 2\ 536$
 $l_\phi = 80$
 $l_H = 160$
 $l_r = 80$
 $l_s = 1\ 024$
 $l_r = 1\ 632$
 $l_\rho = 208$

H 和 H_r 是附录 B 中 B.2 的 HL 函数,其使用 SHA-256 作为基础密码杂凑函数。

密钥生成过程(建立步骤)：

 $p' =$

```
5E1C70F4 512B6FFB 527E1EEF B8AD51DA ED15ADE7 8D69DD26 5562F46D FEDA3D09
EBD6EC7B 7C6FB931 BE416E20 FFCB9873 D3DBF653 85897F39 729913C8 F453556A
D475CF65 3B861673 35F8E4D6 8CF9B2EB FDF1D9BE C5C0BAB4 BE487743 2946416A
CBCD4189 E78316FF 0B69E013 50216582 2E5CDFEA D7257FF8 D5F2E5EF AA5DAE65
```

 $p =$

```
BC38E1E8 A256DFF6 A4FC3DDF 715AA3B5 DA2B5BCF 1AD3BA4C AAC5E8DB FDB47A13
D7ADD8F6 F8DF7263 7C82DC41 FF9730E7 A7B7ECA7 0B12FE72 E5322791 E8A6AAD5
A8EB9ECA 770C2CE6 6BF1C9AD 19F365D7 FBE3B37D 8B817569 7C90EE86 528C82D5
979A8313 CF062DFE 16D3C026 A042CB04 5CB9BFD5 AE4AFF1 ABE5CBDF 54BB5CCB
```

 $q' =$

```
85D9F41E EEBF0F95 CE53ABE8 BE7EDB94 FAF84AFF A04EEC40 8655B3CB 85856A87
20C662C7 B2C4D092 145C8D07 95A7685C 10B1C715 5F366DBF 83C5E4D9 5E311A7E
828692E1 3D5C04CD 0DF4CAFE F743F9EC 181DDB67 A4C6928A DADD9771 D0D5C381
```

4E557D5E C329E8D7 FC527B74 0A3A0BE2 2D6ADB27 FDE1A99F 87CE53B5 3841132D

$q =$

00000001 0BB3E83D DD7E1F2B 9CA757D1 7CFDB729 F5F095FF 409DD881 0CAB6797
 0B0AD50E 418CC58F 6589A124 28B91A0F 2B4ED0B8 21638E2A BE6CDB7F 078BC9B2
 BC6234FD 050D25C2 7AB8099A 1BE995FD EE87F3D8 303BB6CF 498D2515 B5BB2EE3
 A1AB8702 9CAAFABD 8653D1AF F8A4F6E8 147417C4 5AD5B64F FBC3533F 0F9CA76A
 7082265B

$n =$

C4D39A24 A01C9BD2 39EF0B5D A4F7E79D CECA6575 1DCF8EA4 4DECC07F F7F05194
 C963B98E ACDC7C14 E6A808ED F656605C C73CBF69 22EDF311 9EA98591 32A384D8
 E4A9182A FA926596 ACA4B3FC 7F77182A AF08A211 EE1751DC 2C2F8DFA FB4DFC54
 77193020 9E43D11A B8EB491E CCDE1F6A C1329B43 FB9F10E7 D6727739 10317A12
 41DEB3B5 DE753A3F AC3C74FC 24715B04 33C27374 8F59155D 4600D232 A121530F
 6CFC201C 4B6FCFD5 BE0E67E8 8B8B842E 680A07D3 2E9516D7 A60415BE 23D595F7
 19048092 5E04352D 1B2CF4E5 6C8C2571 4AF8B6AF 09AA4DEF 1A9FC5A6 11E2C144
 FE12DF43 3695166C A1F7AE87 296F85AC A628DAD8 829B0CA0 333E77C2 DD761E29

$g' =$

64E6DC61 16767042 2DF48B32 B78491AB 12EA1734 D7EF7A79 86218A86 1A09E001
 D933EFE8 82FCDF1D 484665EC 069C0C97 AE2692DD D78BD52A 6469149E 6A7B7331
 7BD4FA92 6C08066B 7A0AAEC8 6A2A02CB 85C6BB58 BF149AA1 BAA17B41 33361362
 3E8F2C94 22355488 A85314F0 65E7D0C1 00BE5F8B E548075E 79968B2A 72140F14
 D106AE91 3D3DCC92 633AE566 F4B758E3 62989061 40B46D45 DD0300BB 1BFC6D4E
 F26C7C57 93232ADE 3528D8AA 10CD959B 72517C59 AE96D4D9 2F46BA29 FDCD5C7A
 E6B54D86 11AAE7B3 1860E188 26C58B2B E3954022 F2462386 DA24082F 34F47860
 D78F8B3E 66F3BB0A 21FB5671 9FC6D1DA 7BFDCAD6 8FFA311B 9F860A02 CFCF8EC3

$x_0 =$

2822EC39 810EA311 47D1B0F9 3E9C8C39 EA7D2BCD E23AC397 6CE86D02 D9953FA2
 00E67F5A 45A82965 6037E9C2 B477AFC2 9CB15F00 A824F618 449613AD FD037152
 44BD640E F8CAA652 C7761B48 68F57546 F21EFA78 4D7ED26B 665CCB10 97799AE0
 BCB1EA58 9C33B8D6 EFFE4A18 5D20DEFB 497C528E 6ED1CEF4 890B0124 0DDF9171
 179B0462 68700E4F 3BD8DD7A 2D714834 D31ED499 9B462578 3B4BBA42 534E9ED7
 166EF766 3031BAA3 2F02EEE9 33F607D7 4649FE10 BCC83B84 B2EC2584 8F84A685
 A70CE815 0A0A0CEE 292B5F80 C2FEF017 906B09EF 7634EBDA 7466BBA2 71A6AABE
 01E0BD86 E6C297D2 3C355B8D E35BCB72 AAD6B478 F8461F46 DC8D4FC5 210ADA79

$x_1 =$

26107168 30EA074B FBB18879 847AAEAE 15D53982 09005C1F 9FD4D938 C63FC673
 5CECB1C 9C397715 F20D9822 A5FDF667 C1CD67BE 58CA72B7 CB1C99F7 09C4BD93
 8754508D 20AC9938 051A57DD 56731135 ABACFAF7 4D12043F 474F0D85 FAB63E32
 5EB44839 680F1F44 7C2BCA0F 91F5F7F4 929E5DA6 108A0FF7 F4044C5B 4D23B8DB
 7F650442 73F31047 36ECDF7E 04340568 F362E2E1 AF799ED0 5A79A89D 21B301EB
 87957CFA 72C310FD 99108B8B 2F2468E1 215EB6DA B0ED8B72 2C8754D1 C06DCB32
 37BF7D34 FBE15424 20476F1D 0D75867E 63DDAB8B 612911C4 5E174F5C 8F760D09
 D2F70DDb 63948E0F 95BDcD95 2C452932 7D5C7C6D 30B4DB34 AA4294EE A7352255

$x_z =$

20812024	49E048FA	D45C45C1	69CB2224	6F1A47F4	1331CF54	30CC8C95	36C69595
97AFC1DF	B79C6963	EB61BF37	BC094F08	E593593D	43470517	18EB7669	442AC94E
DC1972C7	24355869	844A33AF	F3A03F43	B32A5141	01627BCB	955EFE52	7418A02E
AC6A0D1C	DF0D6C9F	F672A6B1	65FA265B	BDF412B2	DE025E1C	C76E6A98	856B2D5F
23B8050C	9CCCF898	AB123B8A	05EB313A	921CF074	3A50B308	E9027403	A6CEA502
BFCEE2F0	780E7C97	F78BD718	6D378CAD	3308FFCA	16280EDF	8D87F640	D8ED1907
6CB35013	E5A0D0A4	5DFC4170	2A2E13AB	0C86B543	8BCFF6FB	BE2D08D0	6FAB4FBD
AF4CE3A3	6BA4ECF1	845D7DD7	6602C294	6AADE482	1DFC6902	A041DDA7	DB2306D2

 $x_s =$

2DAB31AB	B537B268	4C687EAB	9CCC355B	60B66F3E	253EA955	81C9043D	843DDF13
202A43C1	BCDC8471	850C34FB	18B79C43	6C114B61	D58FCBAA	09F0E8E7	A8FAF52F
31E3CCE3	BFEA1B1B	66758C52	9F50E212	57FDB83E	026E202A	9F16103D	EC40AEFD
77FB6FCF	669050EB	A502FBA0	BD80DA77	DBE758B5	800DEC93	6DEA6A70	10617392
1AA1D917	BD6013C6	C5BE671A	6CAB0138	835DF214	B09D8665	C1F30292	7E6DD6BE
CE71E44F	7048899A	2C822ECB	C21E5E11	03379505	81AB1E4F	43E3921F	F8ECEC9B
D1EC05EB	83701D58	464C47FA	A6927914	B00BCA26	2F76A740	070F6CAE	7849EFBD
72BFCEE5	18B8C7F0	3FEA4C28	06224159	52234AFD	36100C99	5B24F95A	020002AB

 $x_h =$

1484FEE1	A6F872B6	367D9C8D	DDC47203	1B84B266	EADAD7C1	02C901AF	1B8AB266
348202E8	A7D252BC	5350F078	502792BF	40E8F640	A51E0F95	E116AEDD	C3CDF304
6D846297	176D69BA	0A529214	B111A204	E399C908	AFC94D47	D12FC8EE	FD80A7CE
BB0F715B	943F844E	2A0E7647	A82C20C5	DC0444F9	F9413CB8	71DC0DE6	9931FD70
E4E9A917	47515A74	8BFDCF1E	F965F9D4	F1F51B20	B8113012	FF0B4FB9	197483E2
79015452	D716C9B9	47853A65	8D0B2120	E626BDF0	250BDFC8	01A8DF42	32DE93E0
4C055A40	21DC0ED1	18F6879C	5F146580	2C2919BB	6DE5B66D	8C59DE3A	F9E88944
6C3435D6	E6C2CC59	69EFE106	BB74B831	7DEA9EF1	301B264C	30731B74	73718637

 $x_g =$

0C8CEFB9	5A4D8899	EA18A825	7D1657D2	44C31B6B	E324116D	CE418482	D8DC479C
D88B08EC	E085C363	67BFC320	92277C2B	740D3D2E	94F4F946	ADEE423F	319FCC1C
34216692	1BB72869	2C71E0B1	59A5300C	59FFA16A	954F2F3C	54BD58DC	C62885F4
5CB24392	D2F60CD2	3E597D2E	679D4D4F	3402ADF4	A59717CD	E56118D5	2603C99E
3CE610A2	F07C687C	FC546304	5C6D1103	53541407	B2ECC1A9	B6DDE98E	3CA27D62
25A22CCA	15533AF1	DC6B0AAF	555612A9	7D51C784	D882306A	A8029474	C58E2E85
944656EF	9CCEA62A	D5514259	1ED678D6	2FEB41DE	FF2ABAOB	E28D8AD2	7B51C16F
5D9E997D	69C6120B	A07BC137	D8340A3E	2A99CDDA	D4E81E47	38F77C3B	8F07E2FF

 $g =$

B37D2E55	1F0FB3BF	D1D132A6	DE1FED15	4D1B718C	30B0E6A4	A3CAE0FD	DB10D499
F5D63330	59BA3B8A	94E77BD7	6D35AEDF	4EF9042F	CBD54450	6BEBA44B	7447BD6B
EB11C6C9	2194DC11	BDBA2791	7E540972	2AB897D3	2AE3A131	5FE38E96	E7BBDC20
6A25904F	613C4FC4	D73BF059	0CF467E2	020FA2FC	F84C7F93	F93C91B9	C9BDDE03
672F0DCA	A747BA7D	6608F9EC	3D02873B	DB0F1465	1825EACC	03E822E9	722A00CE
807532C2	D3941542	514F4F36	193CC7FB	4211D439	7CF8D9BE	ECFEBD82	37BA2416

F85652EE 88C3487B 90C27443 9220907E D68E60DF 4232994B 0CBC0AF3 B721B21C
 20E001C5 BEB470C6 E5EADFAE 52718308 0C190D20 4E75404C A280B591 B91541AB

$h =$

6E4DE4EB D9B9DBE6 E6869FDC 48123824 445D2BE7 3CA64B9E 085C1DA8 CA19C956
 75B1AD8C E877A95C 4ABD4ADC 7E5687C9 051BE278 CA95CA02 59D7ED9B B4E2161A
 22973A3B 6284B342 64217050 61995D0B B35A9AA6 FD85FFBF 7EEDAB5A E455489F
 2B7D2679 8936953D 63AD4A8F EC000AEF 07761E4D F54CD61E 4C4948FD 7277C7F2
 8CCB9E89 D867245E 684500B8 7CDA3B56 BC381904 A505BBF9 FFE79ECE 308CEB02
 7D3386D0 24CEEA34 54D1BD25 4D5E5FE7 23840F7A A6F38E34 943177D4 EF44D807
 342E989A 622AD67B 03F92BD7 B2A888FA BA5A18DB FC0712FF 89DD9749 DF19B55C
 CC88D01A 037D230E 2A0DEDF5 E53AAA82 E4A3985B 0A3A76AF 2B71A1A3 E7A32828

$S =$

740A7C2E 69C11000 5E984651 52D0C3FC B7CE5F02 46273EC5 0C75285C 8B258170
 72642A07 ECA4A07D CBB1F401 A15B0476 B2667F4C 40202BEB 90655CF9 7E3797FD
 D4E81D82 EB663575 B91E6328 7CB66796 1C1F649B F1D4DF5B 86872A70 0C6B3623
 76B29232 467E2840 3059E2DD 18D4ED7B 9DBA49FE 10AFD6C5 9184DC9E 5735BAA6
 90581346 6F1C8319 FE6C239E E0D6373A 5029582A AC973C9F B264E7F4 596A7898
 70957519 8E98F19B C9100DA1 5D5D8B67 C63D7A2B 141E086E 21385AA3 FADE7EA9
 F7BEA90B 87D62872 E7C0AF10 EF3BB3E2 3C717CDF 41BD2C0D A59CDE40 99C8B49E
 FB4C5B70 1BE3310C 7B248706 DF33D1B2 4596CFBF 86132157 8D05B811 8C5496EC

$Z =$

1DC73C13 2C7B8D0D B4B05912 3A0B9007 4371BA5B AE93E1F9 A6D03A6B E7382F52
 9C01FD94 376D4146 64FC768E 30980362 0D990343 84A5EC50 29A874BB 34BC840F
 A5838DB7 458A4158 CF764E54 997A188C 8CFC9BB8 BEE2D7EC 813D0D9C 80996BA7
 83DB090C 79A3EEE9 E811CDC5 31450218 210C3C0E B4E8A9B9 0B0BA5F5 1F42114E
 9799302B 9CB423AB 6C223BF2 4C7F6CD4 3020A609 148EBB9D 6BC51CA7 24404F52
 79F5AD7C 6C6A377B 3E14ED0D 67CA0C7E 7B72447E CE011200 52A32706 B41D6B9E
 ED984342 5B60AA4B D8774441 DAFF94BA A1906C34 0E9295F7 280B5EA6 EFC0E1E8
 20D9F90D E5F08A8F A7AD42DC 33998790 F4D7BBAC 5E766751 FE3C21B6 A7D5E6EE

$R_0 =$

36F47840 BF894A72 09DEC142 3A8A8A2F 6942EE22 DC6F8163 C464201A AEF98E0E
 C5009F6B 380396C2 A68A8B0B 8E057E84 187787E1 99A0D69D 6A66E8A6 1C8C80BE
 49A2002B F3E7D24A 9890DF46 ECE9C632 36132742 3FE46ABA 5BE5DD9B 583DC624
 E6C9480B A123AD5C 717EC19D 35378DEE E8CCBC56 840DA117 5129FAF2 4A0858D9
 974DDCCF F160D9B1 9A3D006A 3609DCCA E0EB9BF4 728F4D69 7A68B282 D6E24379
 D542404B C76D41AF 7A4E7206 2F43CE03 BF66870B 3B135176 19A048C4 A4D55EF9
 DC2C0981 EC27155D 29494732 417E5EC5 4AB8E721 286DD632 1F765E99 9115F52E
 705BC800 8A308168 191AFE52 30554A97 9893D9BD 8D960E25 4717CB00 4B0C0032

$R_1 =$

A2F6EE15 D0267740 1D2FAF78 09F329E8 4DBA4DE0 3837B8CF 6C63BB0A 4FC009A0
 44000785 32F82E43 02487BE0 765B297D FF344E46 99B6E993 0019BB02 5992212E
 7A3E68FC FD2751D6 5AADE620 A31D0FC5 52BF579B 354F29C4 5F0EA2E9 CD08788F
 6FA10EC2 16206626 3B3FF615 58915213 E259D259 1FF9890C 5CB2E2DD 0306ECD6

B8DC5C6B D8F2E632 285DCC71 7DC49021 0ABE9971 828A419A B27BF5B3 A7004FA5
D6FF78F5 C3992A5E 19A98ED1 6553497C B3AA98C3 A04F887B AB1B78E4 7DE31EFC
D8340D62 7EBD1F8F D22FEA81 460B22D7 86C712FC A5313F55 B355336E 83594572
C0F2210F B5D31680 9C5D3BD7 AD71AF03 B6E0C3E3 FD3AE398 C5118F15 B5E2D875

$\rho =$

0000C510 A08B8603 4D27F9F5 6EAE0EAA 644208EB F7BDE940 2BD8F59F

$\Gamma =$

D857B2D1 92A8EA44 37E8BC76 AA9F3BA8 B59A9E13 F77BF943 A32792E3 5E7CFFFO
EAABF964 742CF942 882DF791 238C464A A3C09794 B411AA9E E9DF212D 6455FFB9
BE820F54 54A15324 EB685348 C09ABF73 CF6AE657 361DAAF0 95A3D1D9 20DA245F
4A86B55E 405BA46F 4FFC2681 96EA5D37 4CA033AF EB40C21D 2582F875 103775B0
130BD921 8EA679E2 E75A8990 1449E939 907BF63C F9CE715B 87E4497B 44B3E21B
A8F4461C 194C350E D9EDEF8F 243569E3 2A3567F4 6EF0EE7C B75F12D6 759E413F
120FFE75 B70AE8FB B59BA2A1

$\gamma =$

8A44F4EF B1679D6A DCE098D2 34996CF8 52D8732F 39BE767D E4A74D7A 7CD2BCE9
A63B5332 5F31B3D8 047376BE 230965F0 31B37355 8EAEA0A1 C18AB66F 04A7D3BD
3E4DB48A 7F931526 184BFE97 6B621C74 1590BBF5 8FC99E27 06B5B164 E25F847A
6D80AC4D 144C47FB 26B45F63 37CBC354 73F6D27F 8FB7B82D 796ACF20 8CF49CEC
6BCCODEE 5DDFAFA3 77835054 84DFC5D7 2474D13F 6D3BF85B 3A7B141E FCE8E665
4723080B 61ED2022 8071E7D8 ED52A149 8EA856D5 556DA256 32728402 663AE107
169C0556 AF2B4D9E 3A3AA0A9

密钥产生过程(群组成员发布过程):

$bsn_I = \text{NULL}$

$J_I =$

A7F56F51 D1C0F7D2 256FDA75 8005E87E ECE23080 276F9B20 96065A47 404486C3
3921EC11 0063BADD 653F9D14 35C742E7 50FD4034 5DEE9C18 720C3775 02F4E1CD
0765160E 5BE01153 9E356EC8 A06CC9D5 D53E1985 EB545DD6 24D62308 D6555B55
F5EC8545 61DBFC8E 042BD9B0 709BCE98 022F2FCA 45627904 04390C09 96B0095A
714609AD B51FC494 97BC3FD1 2BC42CF9 27C1EB04 90CEF1BA 3E6E9EF6 6D074FD0
0AE3A2AB A30F974C 26A8D287 12DC0015 835F600E 6975450D D3563EEC 0380A955
53B0C8B1 C735FD92 A5D5650D

$f =$

00008164 1DE728E6 40EDF561 B2E37604 6D15A72C 3814E51A 7D380AF2

$f_1 =$

00000081 641DE728 E640EDF5 61B2E376

$f_0 =$

00000004 6D15A72C 3814E51A 7D380AF2

$v' =$

0000FF15 694264CF 86AB92F8 89D80D24 1234E15D CAF0797F 0A4CA49B AFDF91E2
3C102A43 887FD7EE 96DC3299 16CACC10 7FE16318 409829B9 0CFA0C55 7F5B5B7F
48CED6C7 8461FC69 7B52E170 0A916A0C B41B9E3F 102CAB4D 334F6975 90E905F9
31A7C841 5BE5F59D 44F29C28 7EFC9BD2 1690014B E9D0928B 12C77AB4 58F8429E

A82420B5 859397BE B719432A FB2E7B37 B286DB8F D3A15167 1CC26834 C04F3CF0
 CE42A1CC 9ADA165D FD80A7F0 0DB1077B D45B5465 35C3DFB3 78C61CFB 5A5932C2
 F56DBF0D A27D7C44 05C1CB0D 05108565 4D33E299 52B176D9 6DD86798 A9D1C5E2
 DF266D49 60F01BF5 1F47113D B8925A30 716F5BC4 6D0FEFEB 299749AA 7378372F
 C167F8A5 7785CED0 BEB86F99

$U =$

BBE3AE81 4D301822 93CFDBCE 8E93F383 7156AE1A 1AA38C6A FB3A905E E8E662EA
 366052DB F759DD03 C52FE609 8D69B666 79C26AE5 BF7E420D E86CF189 2E928751
 E17F1232 0BE0F221 A2E5A4CA 45D4234C CB5E4A8D FEF73A3A FF9F910F C346915E
 F2675DA2 FE1DC06F 0A6915FD 552508EC 6728A0A2 B31BFCB4 97319ACC 1F9C8F59
 DC1175D8 5EE148B1 543417F4 E59F2269 894A9DFC 4F7433B9 80B97591 2B8F714D
 463C08F1 76C9B387 FC7D1C38 E5BA70FC 3EC5BE1D 50F6367E 677AA3C0 BB15BE71
 E11354E5 3CFE0200 06097D6F 10862577 6D4A5E8E E4E0FD70 D86970E9 0512BDF3
 4F828A30 238AF901 6BA60A50 DF230A9B EC645504 D0189730 7F153AD4 C3B1F027

$K_i =$

3704419B 9E4B8439 B804006E D5CF3740 F49D6C87 D7BB6DAA 3074890E FEB287B5
 43B484AE 45C3939F 07547B3E C8AD159C CC2D1361 E4523550 C06630A2 EE4CFC2E
 D84989AE A52209A0 D716648F 8ACEC20E E9AAAFD2 A7D2F992 EF9434E8 8837CCA1
 18957AF1 A77388B6 9EB7FBDC C8A10DBD 213F19F6 190239BD 61FC46F7 FEE8829F
 D7003017 895F9CB6 18DF288E 6F67F5B7 D690DC90 B3432F27 032949D6 B88E8A63
 2491EEE3 F38DF476 D245C62B B901474E 4FDD53F0 A3F9A454 3D1177D4 1493844D
 AAF14EA3 91697E65 A4C83670

$r_0 =$

007890DF 082FD00C DB16176F 6E7CDE5F D085B4B7 0B2A32BE C2ED7379 15238786
 8DE01D77 6921E34F 65AE51DB

$r_1 =$

000A3C90 1A49433B F354E9EE 7ED5A3CD 662441F6 58D0E074 17F930F5 4B332125
 82548ABF 532F7074 D47EE311

$r_v' =$

F82916F0 B76C005B ECCA530C AAEEAB6E 6CC81B49 4707EC20 A21A99B4 9AB4EF88
 FF56F737 532A5977 B9D991A0 7AEA32C0 36EDC5E1 ED1DB86F 2BFDA7E7 EB4EE4C7
 17E06754 5951FEF9 43E86831 04199E1C FC59AC53 50F45B78 AACCB608 4326ED2D
 F525DAFE E005E8BE 7A4EA44B B3A93DE2 ED6DD8DA CC83019B 9E6C03C0 F860EDCB
 6B7386C8 649AEABD 7E1F7980 3B915405 0DE75DD0 88C27677 0CFFC958 CC1F1396
 B223DFC6 22E4DE52 D1854E28 00748173 41C0E9E8 2438F8E4 92B2BCE6 6ADFE6D3
 D4D46B02 602FC0D7 A496F107 91F80B08 CF25FBE6 89B312A3 4DB72AA5 4840CD8E
 C7FACE0C 51E62D7D 7FA16E08 C2A596E9 D38133B4 753EA849 996F0EC1 B926C69C
 1164F27B 14782025 5F6BBFD7 8EB3AB87 92378C7F DEA60A7C 70968961 EAB77B3D
 2BDD08E1 997FD092

$U' =$

9DB5BF27 AF71EC30 63166E9C 83038830 915FDC04 2079E405 4C199118 BE1301A2
 EFA678F1 6C173D53 D6A32A7F C88349F0 CAEB0DA2 146451B8 9A906844 89470741
 6DD3656A 15A96C4B 43C01DB4 CA56B994 6658BF5F 7A4F2DB3 0B9D87F6 620AC8A2

7677C4E3 1306C164 3B65B471 D0361AD0 2EB2862D 1D8AFABB 1915A987 A2C78102
 3E8BDB65 0F26FB2C 5B34C149 731859E3 79E8B552 C56B5835 034ED0E7 E1F79EA5
 D67B6689 37F1C8C8 3824BC07 0F3D5246 84867A00 E520AFEE 72BA73AD 90B3F81F
 32284D81 96E73C66 2719B8DA 949FFC8F BD45E98A 1A3D034D B38E2059 B5D193B5
 9278EF21 53A64B4E B895C8CD 9A6BB9E0 1E1421BF E116E67E 22E291E0 7EA4FBD9

$r_f =$

0A3C901A 49433BF3 54E9EE7E D61C5E45 2C71C665 ABF68B87 67ADD3AB 03A6DA39
 5FB4F211 F25DE84D 94069886 8DE01D77 6921E34F 65AE51DB

$K_i' =$

CDDC3235 1F12CEE9 59A8B63C E23571D5 5136F69A 839EED2E 510B4F05 B1B3CEA1
 6B92FA5C D8027152 4BB1BF3E 6F8C0EB9 B6E9B085 F4745615 5DB85444 72067499
 AFA2CA09 C69FFFE1 2EDBE239 D52C6E0E F5996842 0D6DCD1E F6F6A180 A1D77C02
 180440F6 D0465CAB CFF7D1EE 073C4634 83367F5A 89D2E067 B66757DF DA493EEF
 87810E40 D0047321 4E9ED986 926C182E 6545BBA8 97BDE0FB FB60E6FE CAAB3622
 7E1FBA21 3BD2A75B A2505A4B 0745DC5E A55ECD61 A871BF3D 42AB7E9A 0A16B42B
 CE017E82 94BD5E73 3D2FF022

$n_l =$

D9019EA2 1EE2A92C 34AAB0B2 8A1E77EF C6C9280A

$c_h =$

5A9DD72B 612E1B5C 5EEECA08 F9E82C93 1F889A46

$n_T =$

00005C09 900EC474 B362BC52

$c =$

15F2C5DD 09B2526E C45B06A7 60AB4976 9A1F2AD4

$s_v' =$

F82916F0 B76C005B ECCA68EB 53F6579C 249FF698 DE2E1363 AB4DCD9E 8A2C72AC
 0801371A 36818E15 828C3439 6FE77A45 7F273EB9 367F5EFE 7C6E1935 A096D869
 7B94AF10 42C97D9A 32ABC44C 958BD514 6F81061B 163A2C25 47F42A3C C1C417D2
 9341667B BBAAF86 DD1497BB 5971AEE1 8FCD9E60 A77A446C 6EC31842 4A71142C
 BC415F4A 3C6636D0 983CD57F B1211821 609F1D58 82D57A2E E2585944 2DE2E1BF
 7A5A028D 01E2711D 3B986C1A 56C45A89 4F17E138 46856ECE 9CF71B1D 4ADC0E98
 7442A3C8 F89170DC 3939C3CE A57C6E05 75AA14C0 D1FA3BDB 372D6C5E A64D6E76
 DF6E5182 6CB67274 2C3CC5E0 3BCEF5C2 E452C7C7 D5239132 72400E8E 7B0F04BA
 88D7AC45 B169B45B F7036591 6EE111F1 AC26595F 84FF75E0 933C9E81 02794DF3
 24A5AA07 5A125546

$s_0 =$

007890DF 082FD00C DB16176F CFA22F52 792DD061 477A8688 D33F5E89 1EA82E05
 3BC6CBF5 B828E3B5 31311643

$s_1 =$

000A3C90 1A49433B F354E9F9 96C0BDCB 9D0A9A54 2B0BD0A0 F807F56F 61467552
 A8C8C522 509EB193 9D3E9CC9

$U' =$

9DB5BF27 AF71EC30 63166E9C 83038830 915FDC04 2079E405 4C199118 BE1301A2

EFA678F1	6C173D53	D6A32A7F	C88349F0	CAEB0DA2	146451B8	9A906844	89470741
6DD3656A	15A96C4B	43C01DB4	CA56B994	6658BF5F	7A4F2DB3	0B9D87F6	620AC8A2
7677C4E3	1306C164	3B65B471	D0361AD0	2EB2862D	1D8AFABB	1915A987	A2C78102
3E8BDB65	0F26FB2C	5B34C149	731859E3	79E8B552	C56B5835	034ED0E7	E1F79EA5
D67B6689	37F1C8C8	3824BC07	0F3D5246	84867A00	E520AFEE	72BA73AD	90B3F81F
32284D81	96E73C66	2719B8DA	949FFC8F	BD45E98A	1A3D034D	B38E2059	B5D193B5
9278EF21	53A64B4E	B895C8CD	9A6BB9E0	1E1421BF	E116E67E	22E291E0	7EA4FBD9

 $t =$

0A3C901A	49433BF3	54E9F996	C1365C7C	12CA2437	E6E6B867	D7979EB3	BFA3230A
103FA8D9	71F0F226	5D44F705	3BC6CBF5	B828E3B5	31311643		

 $K_i' =$

CDDC3235	1F12CEE9	59A8B63C	E23571D5	5136F69A	839EED2E	510B4F05	B1B3CEA1
6B92FA5C	D8027152	4BB1BF3E	6F8C0EB9	B6E9B085	F4745615	5DB85444	72067499
AFA2CA09	C69FFFE1	2EDBE239	D52C6E0E	F5996842	0D6DCD1E	F6F6A180	A1D77C02
180440F6	D0465CAB	CFF7D1EE	073C4634	83367F5A	89D2E067	B66757DF	DA493EEF
87810E40	D0047321	4E9ED986	926C182E	6545BBA8	97BDE0FB	FB60E6FE	CAAB3622
7E1FBA21	3BD2A75B	A2505A4B	0745DC5E	A55ECD61	A871BF3D	42AB7E9A	0A16B42B
CE017E82	94BD5E73	3D2FF022					

 $v^* =$

0000002C	84E2EC62	C248199D	9F443C07	7BDBAAC8	86BD4C4E	92A01202	6C0134E8
8EA8D5E9	B09BE372	925A3A49	1608F4C0	C17AD3EA	B200DC2F	7690D690	1C27AD6F
95B3483F	3904D382	9CEFFC8E	F7F75808	A0D66B02	B0F514B4	139887E7	6C195862
C2FBF920	E6E29A2E	A0DE2A4B	ADF53A33	1EA61F76	AAB76BA0	C302CA57	F2877A8D
8DBC6EDD	47D68D3D	3AA0E7E7	3DBF3015	CDB255C4	5C135C09	0F8F27B3	BA640486
92BAEA76	780430E8	223697D6	A4997D69	E6D2B2EF	E70503B7	665EED87	BC7DC7AD
FE4BD40C	CE0E5B72	AD45C8B6	553320B3	4985A722	2A5F2045	813407C0	0FC81E1E
8F856BF3	988EC80F	32390BD5	E284BC8F	4394B9FD	F886CA6A	D04D62E6	D087BA7F
8F686379	68DC7C64	DFEAD7B3	2B25F363	2AAACD91	B48B5D39	15D74EC0	EB9D6B3E
01E8A36F	9CCE6C54	7AF11121	8339C474	AD769D53	EC62C572	57C39DB5	3AD23B30

 $e =$

00008000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
000B7652	6555893E	EFC8FC96	F6171EC3				

 $v'' =$

000000AC	84E2EC62	C248199D	9F443C07	7BDBAAC8	86BD4C4E	92A01202	6C0134E8
8EA8D5E9	B09BE372	925A3A49	1608F4C0	C17AD3EA	B200DC2F	7690D690	1C27AD6F
95B3483F	3904D382	9CEFFC8E	F7F75808	A0D66B02	B0F514B4	139887E7	6C195862
C2FBF920	E6E29A2E	A0DE2A4B	ADF53A33	1EA61F76	AAB76BA0	C302CA57	F2877A8D
8DBC6EDD	47D68D3D	3AA0E7E7	3DBF3015	CDB255C4	5C135C09	0F8F27B3	BA640486
92BAEA76	780430E8	223697D6	A4997D69	E6D2B2EF	E70503B7	665EED87	BC7DC7AD
FE4BD40C	CE0E5B72	AD45C8B6	553320B3	4985A722	2A5F2045	813407C0	0FC81E1E
8F856BF3	988EC80F	32390BD5	E284BC8F	4394B9FD	F886CA6A	D04D62E6	D087BA7F
8F686379	68DC7C64	DFEAD7B3	2B25F363	2AAACD91	B48B5D39	15D74EC0	EB9D6B3E
01E8A36F	9CCE6C54	7AF11121	8339C474	AD769D53	EC62C572	57C39DB5	3AD23B30

A =

B7F5B65D	5D71E0C0	C8A60A74	4C63453F	F4B5423D	352F5994	A74B4121	20868B82
E87ABE07	C1EA3395	1A1FF0D6	9EF7B9ED	55F232E9	CB47E572	2C852D17	9A7DEB66
8699601B	6D4D1DDD	31D6B8F7	7F6ABF95	C4D7715A	78DA9A87	D9F7927B	0DAFCEBB
EF1B431A	1E2FC235	FB6A9240	19EF4D19	A7217279	8FEC6CC8	B0E2ECCB	D77C0F50
A7A3404C	5D6EB115	0ACF6BEA	6971DD6D	17DF3C16	40E8832E	A3E067E5	BDC13BED
6BBACE61	FFC03E4B	91B43F16	0D242E1A	857BE481	8DE063BF	90475B53	0418272E
AF2043C0	7A5B7145	C172C2B0	D14BE91F	11BD808F	2768EC4E	CF0D26D3	DF532AAE
5A33CE9F	F1354085	10D23BCA	E2D64673	ABBE7B4B	6F7FFF9F	2B0F29F7	97AF713B

 $n_H =$

00000F99	66F0E10A	64460CBB
----------	----------	----------

 $r_e =$

09C9F6CF	8CBFA6CC	FA0E2E72	DBB990AF	4839979B	52DE73D2	D44F88C2	08871E12
7703FE19	992BA1B6	76E38940	7BA0B111	106AD846	D913938B	40080BEF	6E1FFA36
83E0C5DA	0F71C9BB	A5EC85AA	1760EB42	7B7F1674	CD884011	16F9697E	502C6D5D
452D1B90	76437A6B	78272DED	F8AFB330	E2676822	4033108E	7BEACA5F	BFA41027
C72FEDBD	B9F14096	18C6F5C4	BF84226F	79F675E9	EC886615	6E917414	342813AC
C74A69B6	1FF0BC48	17F53769	00EEDFD9	0A818FF7	93DF2A5B	657B0C84	6F82D80C
FF8DF797	3DC24969	A7C23B42	432AF901	FF988851	929BDCEC	98B3DFC9	44DD7071
7AAD7BC5	323E2FC7	54234CB5	BF292BFA	8A03F437	D5ADA7DC	42E5DBEF	A4A45453

 $A' =$

936B6F40	16C9D4BB	006A372B	3A79871B	B87B29B8	F43F9335	67CFE0BB	A27095AE
2C415419	06C315A3	C0A966C5	90847A4E	6B7C0FEF	5C9FDE67	9441D9BA	DD2B9863
9F267DEF	A70A646A	5B20E50C	34308A72	0508CA75	24FF4176	DCC865B3	3DF0679B
1C5F561A	1251FB4E	CAA98D09	B2D7C977	1A13B097	EB713823	9A58A985	14B80248
5482E549	4595C4D4	1B1B6E5E	270FC8F2	3B276915	8C404078	68F69173	F41D2076
48130EF0	3F9478CB	8AF2C8DC	B63661C2	B3519A11	F55C124A	121C9A36	D138AD2B
5E9855BB	02945B33	7BA8AFD9	553B415A	68191335	FEE28B3A	6A39A8CF	782EFD79
87241452	B8DF02CF	9A2D6A20	4F5FEB15	6E6D5583	A4EE187B	266C064C	C19D165E

 $c' =$

948F94E1	6ECBC608	091FEDAE	C93C27A8	090EC402
----------	----------	----------	----------	----------

 $s_e =$

276486A6	7A99DEDF	1E7B32B9	7F08967F	73C6635F	7426F178	2D721242	D1C3E71C
44D834CF	C1BB56D4	5677DCFE	D0C590B2	95DE1EA6	1B785F76	80E66819	F1146B72
BE6AAF67	AB54620B	A8F0ACB8	92E644BB	B5F23149	525C4841	F3119F45	9CF1969A
5DBFB80C	A95B7A0C	2C1545AF	26E2A1CC	BE417A37	EC905729	69A98DEB	F4B10D7F
1E016897	FA2ABCC5	8DF5139A	01E59363	F85D944C	A6DDE51B	63E7DBB5	76D1A19C
B301A3A8	5E3F1489	F404DC31	CC551F43	B9DE2701	AA70A4F8	A2B02599	4BDFAF31
C25120F8	68E81114	A8781D68	DC2F7621	17A86854	B284FDE3	908AAB2D	C7186230
49DB87DA	0A3A25BF	12E1469C	9B8A1A5D	42B5C494	8C186CBB	84A09071	9BA1DFE6

 $A' =$

936B6F40	16C9D4BB	006A372B	3A79871B	B87B29B8	F43F9335	67CFE0BB	A27095AE
2C415419	06C315A3	C0A966C5	90847A4E	6B7C0FEF	5C9FDE67	9441D9BA	DD2B9863

9F267DEF A70A646A 5B20E50C 34308A72 0508CA75 24FF4176 DCC865B3 3DF0679B
 1C5F561A 1251FB4E CAA98D09 B2D7C977 1A13B097 EB713823 9A58A985 14B80248
 5482E549 4595C4D4 1B1B6E5E 270FC8F2 3B276915 8C404078 68F69173 F41D2076
 48130EF0 3F9478CB 8AF2C8DC B63661C2 B3519A11 F55C124A 121C9A36 D138AD2B
 5E9855BB 02945B33 7BA8AFD9 553B415A 68191335 FEE28B3A 6A39A8CF 782EFD79
 87241452 B8DF02CF 9A2D6A20 4F5FEB15 6E6D5583 A4EE187B 266C064C C19D165E

$v =$

000000AC 84E2EC62 C248199D 9F443C07 7BDBAAC8 86BD4C4E 92A01202 6C0134E8
 8EA8D5E9 B09BE372 925A3A49 1608F4C0 C17AD3EA B201DB44 DFD33B5F A2D34068
 1F8B5563 4B39B4E0 67E0760E 0243FCA4 50B5FCE4 ED053EF7 9C185FD6 02F58AFB
 D9C6C531 66C3FD46 E1765404 BAEF4688 9E017AF5 F3864268 4764C6C1 6DDA5BFD
 984DD8E9 FBF22B7C 4ACD9334 710E998B 5E9B5BBD 8DBB244A 6B751D50 FF56A0AF
 11B78648 8E943234 0C072A61 B760F81E 3FCAF58E 8F29246C EBF28546 73970AD8
 F97A4F44 80953702 80E71A1D 71F588E8 09D4E412 F8A1C212 1C0E1E1E 0D48C60E
 9D36736F 6CEA1C74 67FCEB89 5B4AD98A 9DEDECC0 EDF48978 72CADF2A D649858C
 9478E8DE B6105EFE 329C4E8C 98FE5AFB D47C9374 93B1CA82 76C76AB6 0AE47C7B
 BA7AFDA0 0E3DC818 E801010C ACD10E1F 20EED483 ADCABE17 CF496C85 F98AAAC9

签名过程:

$bsn = \text{NULL}$

$n_v =$

05737FBA 8F76E72A FC558A5F 0CDE760E 7ED798A8

消息 $m =$

“abcdefghijklmnopqrstuvwxyz
 nopqrlmnopqrsnopqrstnopqrstu”

$t =$

0000967A 2B625B5F 5F3129CE ABF775EA 2E266109 CD2CF2A6 2FA2CFE1

$J =$

945CA21B EABC8BCD C87A1E6E 52AC249B 829927C0 FAE67253 33AC204C 5F11983D
 D128CF3D C6A357A5 550464F3 01F1D063 0A2A8A1C 11D27684 0472FF17 4EBD117C
 378B9591 13723D93 F04B7D8D 7E128168 93BADA28 1D598B82 6F12227F 29C0D27A
 399FDF67 8F240EC1 20D7C53C 9A02DF13 0BD5BA70 51AE30F0 1CD2100A B69D973B
 61A674C5 A2B84D36 C9D054E0 7BE46078 B2317BFC 97CDEC09 E4CD4D8F 22EB8912
 211D6946 031950D4 ECE23F5F 03A487AC C68C5B40 C8112524 6D8624C5 AEB16C75
 5478E2E2 7F215FAD 8BFC48D4

$f =$

00008164 1DE728E6 40EDF561 B2E37604 6D15A72C 3814E51A 7D380AF2

$K =$

77B21B51 859D2D33 19F809CF 20207D5E 3B068C6D 4CE8DC1A 93538179 F509ABD6
 0A4C067A 9F30C3BD 22F5338C CAD6BCC 4CF3BD97 6795B412 DD500B6A AD0299C6
 95C939DA 3A0D10C2 A08646C5 66D941DD 7C8A2E38 D379D25F DBD8BCA1 FD6CD239
 6EFB897C E528D00B 9F4C11A3 181F6141 38E75934 1C1F9960 3461F287 7AF0BF7E
 CA611F21 4B62ABB5 B99C4B9F E803BC67 49C1DAAD 5745C009 B85FACF8 C98376A3
 1FC95709 DC21C8EC 0181F46D 4573386B AC02A5C4 D3A0EB0E C44D525B 6406600E

5F30C50C A6DB57F3 E6FB9390

$r_v =$

0030224A CA1F064B 494CAE6C 0070D07C AE8DC01D A6F61A8E E97E4060 6FCFCACA
 8A46C8CC 898B444F 73E1A0B5 5F37580F 78B6873F 1D76E796 7449AB2A 3B3B3A39
 13457AFA 94623A2A A9A8D25B 657051C7 633CD92C AE4C22B3 FFC2406E E7051B8B
 649959E0 D4EB0A62 4A4C7E6D 5CF87791 2F735F76 A67030E9 E1724BCC A80EC683
 059E3574 D7C1BC66 86F63DCD 4731CB1E 12CD38B3 DC634C36 8A35F6D6 DC64AD5D
 93EC3BF3 B9CBB16C E11148ED C73D8CB3 030385F3 016223A6 392145F5 A8745796
 0489DCC9 EBE6CE7A 6340515A 56CD3731 14B0EA5F B9C113A0 6D46C2E6 9BAD5CFA
 442E502E 7A5C1E2C 301A4208 5925517F 29CB9E3E A97701AF 64B7EA32 2F6237A0
 796EA8BC CBACD0A6 5C9B7989 41696A11 67C09E2E DD0AA8F7 5AECFA81 269AB61B
 CFCB6A15 40685DAF DB5E6E19 AE167723 6EC29905 EF592CA0 FC710D65 6CEF3557
 E5633531 72C77867 D4D815BC DC48CE08 D96B6E9D D31DAC62 755992BC

$r_0 =$

00389F3F A1DCB5FE 13CB0C60 0D69D83E E07071F2 C1EA634F E9EC6401 35AF4570
 F2CC73BD B5CCE8FC 99BFC702

$r_1 =$

00E58F41 A6E84CC4 C83157D0 11A2DA28 1CABD589 7FB752F0 B6D77315 7ADD87AC
 BB1F8C26 77D14D7B 54A8125C

$T_i' =$

3163B89F A928686B 41FEB879 91128972 257E6226 E14A8A2A C674BCEA 7F16A14F
 5744E4CE 7469F754 1C9F103E A21F808B F4EA958D 5AC74C90 94A3E106 6FBBB863
 344A5C67 AE646870 736465BA E5C27D0E F142136E 618843DF D28D3F62 F6CA7AB6
 0032CF6D C03040F5 31B6CE4B 63544838 840BAD38 FC73D30D FDA3178B 64433BCB
 1318C9D5 E377C1EE D9386821 1673A983 E377417F BABBF32E D07DBB35 082869CA
 B11921D1 B0A65913 8A1A9083 0064AEC4 08D8F2BF 8905BE7D EEACB666 F852A060
 E46ACF33 4167967B 26C6FB3E 95F2DEFA D0B003A6 A483B46D 246F175B 1DAD1509
 5F8C5B33 697DA579 6E722F71 DA7EAD57 147F3029 6FD9D5E0 F494CAB7 1EF939D7

$r_f' =$

00006E62 320143D8 BDC28F8B 08E8B2C5 C84D3CDE 303BE097 2E13A494

$K' =$

A4F9D8A0 01BA8A1F F0BC8652 4F446D25 6FE30666 C77ED82B 0D380009 DACA9DD1
 45BF5DE7 D3F1E2DF 23E02917 2DEBA8CB 0066415E 0B3BDE13 D6936459 58D9A421
 DFC404D1 8496C3FC 47A3CAF8 533B45F9 4AF43FA4 B85299B8 8839C450 B59CAEC3
 48CC9BBE CD76674C 0B9079BE 9D39AE3C 800A89E0 32E77F14 3980C178 3C467639
 BE5D7130 E35B7438 0803F631 8E3C61D2 FFB10FC8 3251D3B0 5404F1D2 A33ECBB5
 13139B95 95B66364 28F8C916 00C4480F 276191B5 99020D57 E191DB27 5BEFF430
 B1B1CEDE 676E7CE4 F46375B1

$w =$

0000FE5E 5E5CE52E C1767D74 B2C52ECF E15FCAC9 69D6DCF0 6A79FC44 3980801B
 15C9CF8E 0617711B 46A6FCD1 5A32AA58 945E6F63 49F0FE40 0ECD9008 736FB38C
 D1DDD3DB CA1BE9EE D00F5397 7BD24338 B03A962D A9CD50FD 7F5A41C2 D557639F
 BF9BA20D 2FEC89E6 C60F88F6 4F8D4340 043B5C0A 209ADC6E 611EF98E 66A6CF5B

7A464ABC 6C12F6E0 B1DAA662 2798C528 3E5C4739 05FFC29B 03AA34FE D4ED8185
 0D41DC5E 4FB3A7BF A10F1666 A52566B2 6D5A5789 C696A8ED 3997FCF9 DA8B1A7A
 9CA85659 E65DABAD F609EB28 84F2C83C F60ECC4F 1C5060A6 655F17DE E818AFAF
 8E3CBAD2 D2D15ABC 2AD3A260 D4EBBE8F 3F63A8C5 33E53092 8BC46CE3 1F3FDB10
 29957C6C 65981FDF BA32B3AB

$r_e =$

00000023 D6506BDD E1D3C49E 92169C52 88890F3C 50259C38 7F218621 D1A3B9F6
 6880FDA9 CFA2A7F7 7A15D439 5535017B

$r_v' =$

0001C3BB 0AC68C9C 4FB4A304 B566E253 BEDFBCBC B5BD7AD1 FA42956D 27006508
 0312D73A 3C4BC0B6 978C2C0D BA34D9E5 B7491E3D 2E6E0C23 8F10F74A D7E4DE25
 3582D544 9981EE2E 735834C7 6BD241B8 62942844 66A49529 BDCE5EE6 E7711E99
 9C08CB25 45ADF762 26C77643 D373C65F D1CD02C6 6F0CAED6 8306CE73 1497D96E
 33263435 73A5BFA7 EE62DC49 7265C305 6F49BD6F DC74669C D67F656E 995B1D15
 506197B1 9EC66C1F 29CE909C BAD3DFFC 2E4CCB41 74A88E6E 69B79B50 35712D9C
 32E1BE9F E906C245 2FA66081 82E9A005 AB52D8EA 6CB83712 88678DF0 2291BB63
 9806DECB ACB0F055 3F25BAE7 295DCFF9 D55FC2A2 B0FDF6AD 136A0539 C3DB754C
 00055648 67F6AA2A C7F37EB1 4CEA3229 BB976A4D 77C85682 1B6A3820 5665AD28
 B1EC867F 940EB08F EA1515A2 4059F409 FEEB4D6D B2A74597 F457544B C2E0E7AB
 BA73CBCB E10A1E01 1EBB912C D4C4E747 4FA388C4 2399655F

$T =$

35AA6B44 2F55CE26 0E96A84A AAE70D08 6A255808 848B1910 DF29EE41 66EE180E
 3C6BF535 3C0A295D DBFC89F7 EA9FB726 144C6A2E 4E926446 CF9A4AEA B77B4155
 43F2430E CE5A58BA EF2F4A4A 92EADF6D 208245B4 83FCE098 0704DF43 325C900C
 1D732814 67E4501C EFC453C8 7F881551 F94DE3F1 631DA3F8 9E20187B 2D6F4E0B
 5FF1CE6C F3E98A1E 7D9C44B1 73868078 47A03B0E C47ED9A4 BAAFED04 8AA6E27A
 B9F5A36F A2E22207 D3651DEC B3DFF319 39012F61 F066EED6 ABDC41B7 F9FFEBCD
 33A79C20 19055C96 CD6C5943 419C0342 9B715969 C2E19938 D0B3632A CAAC46D6
 DA9AB048 9C050DFC 9C35DA5E B211A3B9 07DAF02A 1807023A F24287AD B2A0585F

$T' =$

11302F73 54D6503E B7A68B19 BD9A13F6 51B016FC C485B99F DE0F66BD 167C0F3B
 FDC1773B EE6D671D A0873A32 045EA7BF 9562B010 42F908DC 70B3E867 C88DAC0E
 B045A29C 0C563BFB 7CCBEF1B 95D79517 C10ABDE2 CA9345BB 55219085 B06321F5
 C5E99FD9 44F00C04 0ABB35D0 A3021ECC C19ADD26 B9093ECC 1D8963B3 C7644A82
 49CF48AC F849051F A67631D7 F00EADD0 0BBB8941 CAEAB5F9 11E62437 906701A8
 09048730 9B065900 9413FCCE 6B3BA139 F649E498 4D492728 3EB439F7 5A0BDF8F
 D2CF8B84 7AA6C6E7 1B7574F1 48A18A69 421851F0 625EC66D C7BF1E21 0F80A7AC
 46A65E58 A5C19077 F1E7D109 4D23EE5F 5EE433FE FF836AEC 0BD2BA4B 19A9007F

$c_h =$

4C51D9DB D35EED8B CBD3D7D8 C629477B 9C7A1042

$n_T =$

0000C450 3D3CDEE8 D1976338

$c =$

59D94EA3 E9AD8AA8 C1407819 FE6F411C 6B611908

 $s_v =$

0030224A CA1F064B 494CAEA8 8D1557F8 9E37E212 3B31AE7C 182E935F 64632267
 069B126C 2891E753 393FCF63 89446561 F15E872D 1AEBAAF9 AB6620D4 533D02ED
 2A75DF3F 2C9C4CC2 601A6072 75B8C2E4 FA6913AE 7A2E134E F0D0231E F589958B
 9437B7C5 69CF12E4 721B21FE 5FB5CA6C 62EA7D6A B42C8D70 63C144E0 E7296A66
 C91DFA5A BCC4B913 A8DCD9B2 2579E7AB 0B1C004E B7214037 A1FEB235 597F8797
 5B981985 EE6A13CF E1572153 B65B4C9F C3B88D86 9412C7FA 2017FDEF C1A3EC6F
 C7F67D0C DF5DC48D 2B2F7338 1D6DD6BA C1354C35 FEACCD A5 1E0184D8 D98BBCA9
 735C1729 E19717CA 075AFE1B 16812439 84DC93FD 78023FF1 3D2111AE 4E8C2C8D
 FA036652 4F481D15 1A4955F0 0C5F87FA BCDC4FCA AFB62DDA 6C2C1EB5 367BF648
 77BA9F4B A4988AB1 C37CDA9D 856241B2 AAA26AEB 62A1E654 A33BEB8A 8F6A8CC7
 CAF61C2D 72DD364D 331C3A85 F8073786 F5187F00 92C15FD9 85858A04

 $s_0 =$

00389F3F A1DCB5FE 13CB0C61 9B1832CA 0C6CFC0F 0F9E178F 762015CE 6943A11E
 09482AC2 2A0A2B52 4843C092

 $s_1 =$

00E58F41 A6E84CC4 C83157FD 7B46DC38 3DB22F5C D70753E5 865BC17D 2B93F908
 FECFDB64 8A615473 5C2BB40C

 $s_e =$

00000023 D6506BDD E1D3C49E 921AA22F FEF9941 46635A0A B8185795 F0DD107F
 5F39F279 C5F2F6AA 00FB3FEB 70D20293

 $s_v' =$

0030224A CA20CA06 54133B44 DCC9FAFD 26FB66EE 06584D12 3171A332 2B4DA285
 09C94131 5DCEDD69 793152DB 87AD3505 27299385 2DEA266F 1E57D053 7805EC87
 29845715 C4F694E6 321AD591 FC18E269 48BBFD1F 1418E217 582BF73D EA6B9397
 C45D1C98 C99763CD 50A17AB4 19C73F5F 27C65295 DFDA3409 C8DA4628 54DF3AEB
 1C5AB84E AC1B03D9 CBE1F6B0 74AB48BF 8BB09661 789EC749 1026989C 7D51E3E0
 71AE8131 A96DB0D4 7EB31519 E5992D7F 2DE7DACC D1DC5E7F 6E5BB1F8 475D12D6
 8ACA9817 58851D81 47912C86 FD1D6758 0A1B1B56 7398F763 6EC114D5 96FEE171
 927804E9 44CCBD66 37003A85 7A75C79B 066CB07B 0F77FD79 324FE8CC EBF41A8
 57A23304 0A474B3C 79A6694C 720F1369 C70FB7B9 27FB7D87 1EB8DE4E 814CF842
 03147D8F CDF22B43 6CF72D44 81E1DEB2 52EA2329 A5AC721B 4D21ADB2 04F45E61
 387225B6 A664B686 6EB4E794 250F1692 580C2961 BCD91BF8 C8CB6C5B

验证过程: $t_1 =$

00002CEC A751F4D6 C55460A0 3C0CFF37 A08E35B0 8C840023 D6506BDD E1D3C49E
 921AA22F FEF9941 46635A0A B8185795 F0DD107F 5F39F279 C5F2F6AA 00FB3FEB
 70D20293

 $t_2 =$

E58F41A6 E84CC4C8 3157FD7B 4714D77D 540C12D5 1B1EF1E7 F6D9AFF5 A066050D
 DF797C19 D774892A 94F7AD1E 09482AC2 2A0A2B52 4843C092

$T' =$

```
11302F73 54D6503E B7A68B19 BD9A13F6 51B016FC C485B99F DE0F66BD 167C0F3B
FDC1773B EE6D671D A0873A32 045EA7BF 9562B010 42F908DC 70B3E867 C88DAC0E
B045A29C 0C563BFB 7CCBEF1B 95D79517 C10ABDE2 CA9345BB 55219085 B06321F5
C5E99FD9 44F00C04 0ABB35D0 A3021ECC C19ADD26 B9093ECC 1D8963B3 C7644A82
49CF48AC F849051F A67631D7 F00EADD0 0BBB8941 CAEAB5F9 11E62437 906701A8
09048730 9B065900 9413FCCE 6B3BA139 F649E498 4D492728 3EB439F7 5A0BDF8F
D2CF8B84 7AA6C6E7 1B7574F1 48A18A69 421851F0 625EC66D C7BF1E21 0F80A7AC
46A65E58 A5C19077 F1E7D109 4D23EE5F 5EE433FE FF836AEC 0BD2BA4B 19A9007F
```

$K' =$

```
A4F9D8A0 01BA8A1F F0BC8652 4F446D25 6FE30666 C77ED82B 0D380009 DACA9DD1
45BF5DE7 D3F1E2DF 23E02917 2DEBA8CB 0066415E 0B3BDE13 D6936459 58D9A421
DFC404D1 8496C3FC 47A3CAF8 533B45F9 4AF43FA4 B85299B8 8839C450 B59CAEC3
48CC9BBE CD76674C 0B9079BE 9D39AE3C 800A89E0 32E77F14 3980C178 3C467639
BE5D7130 E35B7438 0803F631 8E3C61D2 FFB10FC8 3251D3B0 5404F1D2 A33ECBB5
13139B95 95B66364 28F8C916 00C4480F 276191B5 99020D57 E191DB27 5BEFF430
B1B1CEDE 676E7CE4 F46375B1
```

E.3 机制 3

安全参数：

$t = 256$ 。

群组公钥参数：

G_1, G_2, G_T 是由在 ISO/IEC 15946-5 附录 C 中 C.3 中规定的 BN 曲线构建的。设 q 是一个大素数。 G_1 是椭圆曲线 $E/F(q)$ ，其中 $E: y^2 = x^3 + b$ 。 p 是曲线 $E/F(q)$ 的阶。 $h=1$ 是 $E/F(q)$ 的代数余子式。设 $k=12$ 是 $E/F(q)$ 的嵌入次数。 G_2 是六次曲线 $E'/F(q^2)$ ，其中 $E': y^2 = x^3 + b/\xi$ 。 $E'/F(q^2)$ 上点的总数为 $p(2q-p)$ 。 $E'/F(q^2)$ 的余子数为 $2q-p$ 。 G_T 为 $F(q^{12})$ ，其中 $F(q^{12}) = F(q^6)[w]/(w^2-v)$ ， $F(q^6) = F(q^2)[v]/(v^3-\xi)$ 和 $F(q^2) = F(q)[u]/(u^2-\beta)$ 。

在下列数据示例中， $P = (P \cdot x, P \cdot y)$ 在 G_1 中表示为 $P \cdot x \parallel P \cdot y$ ，其中 $P \cdot x$ 和 $P \cdot y$ 是 $F(q)$ 中的元素。 $r = (r_1, r_2)$ 在 $F(q^2)$ 中表示为 $r_1 \parallel r_2$ ，其中 r_1 和 r_2 是 $F(q^2)$ 中的元素。 $s = (s_1, s_2, s_3)$ 在 $F(q^6)$ 中表示为 $s_1 \parallel s_2 \parallel s_3$ ，其中 s_1, s_2 和 s_3 是 $F(q^2)$ 中的元素。 $t = (t_1, t_2)$ 在 $F(q^{12})$ 中表示为 $t_1 \parallel t_2$ ，其中 t_1 和 t_2 是 $F(q^6)$ 中的元素。 $Q = (Q \cdot x, Q \cdot y)$ 在 G_2 中表示为 $Q \cdot x \parallel Q \cdot y$ ，其中 $Q \cdot x$ 和 $Q \cdot y$ 是 $F(q^2)$ 中的元素。

$p =$

```
FFFFFFFF FFFCF0CD 46E5F25E EE71A49E 0CDC65FB 1299921A F62D536C D10B500D
```

$q =$

```
FFFFFFFF FFFCF0CD 46E5F25E EE71A49F 0CDC65FB 12980A82 D3292DDB AED33013
```

$b =$

```
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000003
```

$\beta =$

```
FFFFFFFF FFFCF0CD 46E5F25E EE71A49F 0CDC65FB 12980A82 D3292DDB AED33012
```

$\xi =$

```
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000002
```

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001

$P_1 =$

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000002

$P_2 =$

E20171C5 4AA3DA05 21670413 743CCF22 D25D5268 3D32470E F6021343 BF282394

592D1EF6 53A85A80 46CCDC25 4FBB5656 43433BF6 289653E2 7DF7B212 BAA189BE

AE60A4E7 51FFD350 C621E703 312826BD 55E8B59A 4D916838 414DB822 DD2335AE

1AB442F9 89AFE5AD F80274F8 7645E253 2CDC6181 9093D613 2C90FE89 51B92421

SHA-512 作为基础的密码杂凑函数。实现特殊的密码杂凑函数使用附录 B 中的基础构件。使用优选的 Ate 对作为基本的双线性映射函数。

群组公钥：

$Q_1 =$

13B9155C DFDAAA36 2264EDC4 CDCCEE31 62D30ADB E7560ADC B79AE0EA 789E0197

8CD127DF 3D171922 C96D8A19 C0F1E043 FB40E88D 5A3D205B EE0075E3 3EBD7BB1

$Q_2 =$

ED1BBE1F 064A969A 34DD5193 0B308272 B71D1066 B9F01F0E 960BBBC1 CB89C72C

EC62B0B7 E760884F 99E07FED 478EE679 2E90E531 78D7A334 2004D3AC 906771B0

$W =$

DFF0F697 65D4B585 32E96F63 BC4F89B1 6B45A77F 877B6CDC 33B42129 148F0A0B

A2B632CA 53E96E80 02B51788 F209285D 1DF7CE67 8AA5AB37 BFD9767E D9B5CDFA

F31DEA38 6566E809 8D08E07F 1F3E2B13 A081442A 034CDAB6 68408D80 FE8A0E85

D48059DC 02B7243F 77650281 B8527C14 0249C560 9588D762 23BC464A CA24062D

$T_1 =$

A88E9AF9 251298E2 C3612EE8 D6A67716 49047569 D1832D3F 2A79B69B C91D0390

2AD8119F 2636E7E9 3A054C15 4993DAE9 D05AE48D 8AFA04F1 208456EC 3C27195C

F1AFBFF6 0E58842D 9411F4B5 F41451B0 90461A81 EDCF9166 58A6363A 52185AC1

084C99D3 DCCE7FCE 78E03887 32F1803C 7B67AA6F DDE0FCCB D0B03A59 522A84E4

F84AFF50 A065C4EE F49CAA34 46F9D26C A1617149 32258454 9044BEA4 0BF7FE26

816373F7 2FF2FA24 52A4D94C C1A7A5C3 0336139B 164516CB 4B9938F3 6DC87EAB

B353DFB6 82601211 36690E05 318ECFD7 3F32E795 841DC8B5 BE49179D CFA95A2A

C41186E8 6C0256B0 252FA006 B362B211 AFBFA4E8 616485FB EB1CF1BC 2CAE1051

16A6C0B3 868E6D79 B6BDDE1E 26064665 82845A97 D3B79378 6B9D1433 94433404

45D147D4 2F17CFF1 DDEA1152 AE01883A 10EE5C16 CDB548E9 162C70B4 1E1938E0

18E9AEC5 DA74412D 70076037 2766F700 BB7951F3 7C8A2BB5 696E101F E00A5EBE

B44E0E02 59B5CB4A 6A868BCC A213A0E9 F25CB023 B215F9BB 43C154F4 C8AB16A6

$T_2 =$

EA09DB32 1E161D34 D8AE05D0 E6440E78 175E33D8 05A8A979 3D4D3628 997E0794

696E6F2A 99D45E05 F7F84CC8 5749ED15 24E72483 128EE61E 44BCEEB7 901C81AC

B245C825 8808DA12 0DE7D923 0E0E7AB7 A718B25E 1C620095 4304119E 8DD9B5CC

0EC8AE34 766D6CE9 EED9C35C 913BF832 EA0B7D5B 558DCB73 91B4F060 1124A223

BC52CBAA D25F9B10 E9BC87C4 8C1A0BB9 75C31AF0 2C29ACFF 45C14AED 7D7846B3

CDB7E64A 917DDC27 783AE81D 28AFAF40 CC6ADE60 12E11C9C 21AE1854 4335410B

5D148C3E 1D397334 437CD6A8 53E692AD 354484CA 87EF86E6 A4A594F1 D9C26426
 B9061BC8 73B0C390 0005B629 31C9FC1C D48F78B5 0BBEB79D 0ED8C2D7 E5926BFD
 8F6B9F3B D37FEB6B 859CF998 FDC77376 87839B01 56658267 5C062664 FCB1CC4C
 5F7B3D4B 86474FDE A6EEBED3 362FE562 66B50DA9 B4193E17 CD2E0726 59F1556B
 97C6559E 099E05B7 618770B6 EA75498F 016A06E4 5255B296 88C727CF 59AE9B8A
 94763F4B F89C4615 23DB6FCD 210FDE6B ADC58BCB 48433AD2 F4B7C5CE A2B311ED

$T_3 =$

77D07370 3278E443 F1F61001 A0E1534C 833E9493 37D37F1F 6D627BE8 252B9212
 6C176BC5 2C8223CB FE6CCBD5 A32F0591 62FC0882 5DB7E214 65B25F37 4285E6D1
 A20C6F53 4720821E 9F3C6111 08FDB364 E7A53A32 AD67C386 E9B8F822 66812D87
 B190FA69 0B5E29D6 9A4A3329 C3DFECB8 0E77C08B A66B1DF1 C3981DFB 2728D030
 B3E0E570 22DBF9A1 93C67759 237CCEDC 44B68C83 013EC75E B60B08CE 8391F3B6
 12518762 34B202F9 89DE6287 93488CEA AFDB133F 4902C350 6398AD0C A1DCE367
 5DDBA3C6 9C7C32DC C785BBE6 B70762BD 3C50A989 82E6AF3D C9E69F5F BAF48BA6
 910EFA01 2AB49848 5C8CODE2 F44F04E1 F35CF786 1FF5D220 D6A37EA0 8C3D867E
 642B8033 5581AD42 F9E0DA36 EE1196FF DACF211D 6DF20422 4131EE3C AA8B54B9
 D61E662D A62D458A BFCE38AE 24899DB0 87B37DE8 63EBD2A3 9F06431C E0E71680
 98A3D7AF EA21574D 7B806D1B E6EA99EB F4D96589 DFE34CE2 CA605E44 71789970
 77E0752A CB248031 F43CDFD9 C2B82A28 88DE9DC5 1E457D71 E4BDC999 A518B75F

$T_4 =$

10306554 CA58B236 43AE81B7 0DF214EF 741D822F 443076A1 DF8F266F 1EF8053B
 21F605E4 C9C4B77D B33D4091 26440477 F147ACC0 66BB02A6 5ADBCCA2 8DA5A4A3
 8EE50EAE 0906020D FF213F98 C8157452 474D8074 8AFEAC3C 86B7C879 A854FE7E
 1C97B19F 3215C4CC 4AE5CC58 83999BDC D75DE341 E40EB4B9 189BD1F7 9EB777E3
 503522BE F06CE9B0 DB028D36 A101A4F3 9ED6524C 74916018 EBB2D391 EB709F90
 3127658E D21D5E2E C79EDB84 153274B5 7C9F8E30 9342F54D 25966FCC D5B65645
 3DA22F68 BEDD7023 8EDD2439 95B3345C 3DB9CCDB D805C1D1 67F08CE2 4F52108C
 D16D4D75 476DC0E4 412353CB DA41FB8B BD6991E3 62AE1954 6CE9DFE8 79E6A669
 DB9EACDE E9DEF5F2 97145994 7857F84D 43C8C9A1 EA2AE009 E571606A 4F66295A
 3D050A3C AA4E4E59 12AE1781 33F816EE 67EAF1D3 2D129ADD 3C382CD8 95CB163A
 6E5AFE63 21D17C65 48D42905 444F8736 B8F2605A 5CA9496D 4FACB65A C9E50162
 750A7FDD 1D3583C0 D44541F8 3ECF1FEF EC1E435D D4573418 82638390 D7A61B66

群组成员发布过程：

$y =$

4484A7A0 B04BD4C6 A5D22451 96BFCCED F0A256E6 15C05E70 29C1058A 5B5EF4A7

群组成员签名过程：

$f =$

59A5DC1F BEEED102 D3D17F7D D6BDD1B3 B7C2471D 7E824781 477F9297 E9F1ED58

$A =$

99F75E90 879DD188 C6866E61 85F6C5FD 8EB9FC68 4DFC40C5 36C8EF7E A7FB11F0
 700E5ECE EB64A2CE A0DC73F7 379E5B80 39443B70 0C9E5DA6 EE5E0614 A784C150

$x =$

9EF27558 0B86DF21 30E65A8B 10CE5AED CA8D285E 5665BEA4 D7A18DCB 14FD26FC

群组成员发布过程：

$n_1 =$

0D5E27F6 7FD0BAE7 B4558495 0B7E9241 C526897B 483912F9 E163D41B 760CD838

$f =$

B1B1CAC2 285AD778 7BE67201 37311A75 5BA167A2 367FE4DF 39A9A119 DBEA455F

$r =$

B35ABB63 E0143D81 254A87D5 29E6B4D1 1466F1C5 AF6E30D9 0E8B5C92 4CF6E1A3

$F =$

57E03F20 F9A9E164 0B5E7DD3 FCA4E8EA 71CBFE81 C63C7347 B8C4A102 34577B6A
4DE234F6 0A86262A 25B7F1A6 D7D24C6B A282C04A F175F935 1E9702AB EF2458EF

$R =$

BAADC096 476533E3 3A81D123 D1C90281 35B71732 D0573597 D5C22656 BCA5E71D
47B50B1D 02833BE2 A1480713 8E0BECA8 90678107 683C5920 28108D75 EBC07CF2

$c =$

A1667B92 7F17EF38 1498A4E7 FAC2397B D6D12CB4 DAAD59B1 1F9BCB9B E5E8871A

$s =$

1B187D33 E4CE10C9 0C4AA6E5 EF5742D5 C0B4BA7A A743144E 7427C90A DC8A19AD

$R =$

BAADC096 476533E3 3A81D123 D1C90281 35B71732 D0573597 D5C22656 BCA5E71D
47B50B1D 02833BE2 A1480713 8E0BECA8 90678107 683C5920 28108D75 EBC07CF2

$c =$

A1667B92 7F17EF38 1498A4E7 FAC2397B D6D12CB4 DAAD59B1 1F9BCB9B E5E8871A

$x =$

263FF7FA 973F78E5 51863EC8 F392FBA0 71662D5A 0355C214 422BC4ED CA0D21FF

$A =$

DCF4F117 DCD4F1E9 187709BE 9DE95870 091E0C78 9D3C6149 A8DAF221 F37F8FEE
D5B731A3 693C0F17 D14D05F2 8F59605A 0872A1D3 D2055801 DE02BB19 5A7823C2

签名过程：

$bsn = \text{NULL}$

消息 $m =$

0460EAEF 6D8324EC 4D04EC06 42B1EE33 BDC66299 9C39453A 6CE7AB32 CA00EAB1
ED60AA91 4125EBDF 3846C71C 26EE64F7 D49BFAF0 510F2131 87728A83 C639BFDC

$J =$

088AF78B 3EF3F55C C0CB3D6E 29E7DA58 2EA35EFF 8DAC4556 45509925 73E40926
3AC4611D 126147BE 06632127 05B7196D BEC0DFB9 4C2363FE 784D2236 C8D7064D

$K =$

B3187052 CB49EA57 AD88018B 2D763FAB C4FAF474 9EEDD937 8D351D5A BCB69A61
2BA4EA00 2F09B0AC 18943F24 A8DDD28A D9B6652C 4EC77F4D 605BFD7C 056BCAAE

$a =$

BC907239 6207C3EE 7D7963DC 557D5C48 B421BA83 63F7EEEC 1ADC91EC DBC20B87

$b =$

1CE910C9 2A6E6089 6FFFDF5D 6C6A539A 43E8B9AA 2CC2D8F3 777A4D38 BCEC38BF

$T =$

99C83125 70E7B0FA AEDA0C52 2DB86180 B47B9ED0 0F522BEE 77231F78 F7E0B877
EA63DE9B 901D0628 3F12CE61 E82C4889 DFC14885 746CAB3A F03B3AE2 86D0AF4C

$r_f =$

5D6654EA 623F76D4 C66698C6 445E5052 738A0D4F 67282065 428A20B5 75462809

$r_x =$

BBED75EE BD0E36CA 68AB0381 C0B7FEE4 5CE8DD1B B76D7DDB C4E3A317 17DC9784

$r_a =$

D4C0E43D EBD219CC 31FCC123 A5170CEE 05B88386 9861081E 82CDD950 2EE7ACB4

$r_b =$

93D2970F 7816C3ED 4B431626 76613E4F 764518F3 977E5A51 14D4943D F9394A81

$R_1 =$

F2AE4446 6A998700 F64E628C 877F58FC 71260C63 CBA058FF DB3069FF 2E0328C3
00DD2A5B F9D0EB4B AC25C9F5 09F86F02 7E97B0F9 FF86635E 0776B634 726BCBD8

$R_2 =$

4B80317F 1F383E10 3248CAF8 C9F17415 9DE322E0 0B1312D1 7195350A 9EF3CEA2
6F3E0F3C 421F7B87 D51BB83E 3ED9D425 81C66598 1BCBFEB1 9FE7A830 70A24A50
42BB1F77 7477C95A 78247A5A 1F559C69 93C2C188 CBC8BC1D 8217E418 622CC8F0
16964584 AF3F46D5 B888FE98 C2F23DCA BAD76801 7C31FED2 DDFCC8C0 6154DDA1
565133D3 FA53406E 254A42C3 7DB02DA9 E582B8A6 8E9C13BF 895DC044 8A39DBF6
C8ED4EB6 2E62314E 43D2D2BA 48EBACE6 5F1B911B 1E0FC5EE C052F41A 024C71E7
DD56A764 1F50BB11 809EA9E3 3D0C564F 538A924B 228AE287 E936F368 695E392F
E4FB0577 DA1381B4 7427BA11 E5FCC0B3 4A77777A E75E08B8 9AE35C66 279E925E
5959C456 885E60AA E91CCEA7 F9BCBCDD A6552B89 28A5CBB6 26C319A8 23710080
3C881E86 DB8BE86B E1F0DE10 CF95924E 9CCF3880 09F1C9F0 BB961783 E78691BE
0F8272ED CD936ABB FB8B9EAE 29A685F9 815F0A9D 4682F299 9B027024 8EB9A659
C6F4728B 31FC3E16 3717A259 F6214594 D14A3FC7 E5BFAF64 93AFE578 ACE8CA34

$c =$

CBA0D828 1B49750B 3AE9601F DDF83481 CA848EE5 2150A9C8 CA189F44 BF7559E0

$s_f =$

43CF67A4 80EDE9B6 26327E7E 3E59D4C2 C1D80BF1 68817C32 6E4BF694 F8C5E940

$s_x =$

A7C8EE59 BE1C511D 45FFE1AB DD37D978 A4119EC7 ADDBB5A2 B7F693A8 040A1CF5

$s_a =$

4962D9B7 29F06A62 60841392 9530F407 A94341EB 223831FA 67483666 3BF5AAF4

$s_b =$

1F00F7F0 F7C4C0AD 400A5281 51BB6DBB F7FF24A3 78C0BCE3 21FAFD8F 76CACC1D

验证过程：

$R_1 =$

F2AE4446 6A998700 F64E628C 877F58FC 71260C63 CBA058FF DB3069FF 2E0328C3
00DD2A5B F9D0EB4B AC25C9F5 09F86F02 7E97B0F9 FF86635E 0776B634 726BCBD8

$R_2 =$

4B80317F 1F383E10 3248CAF8 C9F17415 9DE322E0 0B1312D1 7195350A 9EF3CEA2

6F3E0F3C 421F7B87 D51BB83E 3ED9D425 81C66598 1BCBFEB1 9FE7A830 70A24A50
 42BB1F77 7477C95A 78247A5A 1F559C69 93C2C188 CBC8BC1D 8217E418 622CC8F0
 16964584 AF3F46D5 B888FE98 C2F23DCA BAD76801 7C31FED2 DDFCC8C0 6154DDA1
 565133D3 FA53406E 254A42C3 7DB02DA9 E582B8A6 8E9C13BF 895DC044 8A39DBF6
 C8ED4EB6 2E62314E 43D2D2BA 48EBACE6 5F1B911B 1E0FC5EE C052F41A 024C71E7
 DD56A764 1F50BB11 809EA9E3 3D0C564F 538A924B 228AE287 E936F368 695E392F
 E4FB0577 DA1381B4 7427BA11 E5FCC0B3 4A77777A E75E08B8 9AE35C66 279E925E
 5959C456 885E60AA E91CCEA7 F9BCBCDD A6552B89 28A5CBB6 26C319A8 23710080
 3C881E86 DB8BE86B E1F0DE10 CF95924E 9CCF3880 09F1C9F0 BB961783 E78691BE
 0F8272ED CD936ABB FB8B9EAE 29A685F9 815F0A9D 4682F299 9B027024 8EB9A659
 C6F4728B 31FC3E16 3717A259 F6214594 D14A3FC7 E5BFAF64 93AFE578 ACE8CA34

$c =$

CBA0D828 1B49750B 3AE9601F DDF83481 CA848EE5 2150A9C8 CA189F44 BF7559E0

分割的签名过程：

$bsn = \text{NULL}$

消息 $m =$

0460EAEF 6D8324EC 4D04EC06 42B1EE33 BDC66299 9C39453A 6CE7AB32 CA00EAB1
 ED60AA91 4125EBDF 3846C71C 26EE64F7 D49BFAF0 510F2131 87728A83 C639BFDC

$J =$

1A0A7ED3 F9381C38 FB34ACE5 AB735AF7 9E35E06F 88EDE2FA EE34B595 249C06C1
 BC0C71D7 F707890A 4F39409C F5FD6A0B A00CC0F1 4B432C85 EA44BF39 A842093B

$K =$

C3CB5651 BB35B629 3629E0FB 856FF712 9705D791 D7333CE1 8858C4F1 1ED50703
 2D49F1DC 0B3F68C9 40EB9579 C02C6702 41127E04 3AD710AC 5726A949 56F04BD0

$r_f =$

7E30ECD8 86B53233 EF7403B4 8CEA199F E950B959 A6EEDA7F 87DD2350 A482F637

$R_1 =$

9A70F54F DCC81B2D D4D67683 678E1219 7B2E72DC 5ACBDAF4 51C6C547 80AB7341
 041E9FDE 4FF5E949 D9DBCFF7 CF33C2BD 72809B60 AA63E4A6 2C634181 0C2D871D

$R_{2t} =$

604FCC1B AA861786 F747635D 471C69E6 9EA5C45D F6DA2E1D 1E8DDF41 4854DB2C
 4EF8CF5F 1BD5EC1B D13EC007 A5740FED DA8152C1 0521D2C3 A004C1BD D72057BB

$a =$

742B0A9F 38F23E74 761205E9 79059E6B 0FE9503E EF75B0FA 2C5EA463 C12422E

$b =$

2991FA0B 4845CAF3 1F9ECFB5 9D0286C6 026657DE BA8351CF 1921C523 A6ED716B

$T =$

79CD7C12 E7674B03 51FADD06 4BC439DB FEBAD3E3 4D656D68 725404B1 58D03420
 B82538CE 93C37B50 6652F862 DD172F10 E5F5A6B3 C4ED20E2 509B8F0B EEF147E8

$r_x =$

11547E69 7DAF14E4 416E0649 2A333385 40D07E84 07A25821 A42E96E2 FC535DC0

$r_a =$

8B0609B6 0A131D79 AFEF21D8 03E5597D 4B0BDE83 82EBF786 EA023E77 3A9EDA5B

$r_b =$

7A2A8F3B CD46D6A4 DB1A9F2D 25EBF959 01AAAA53 8E88B7E3 A1D2CF79 6718728F

 $R_2 =$

634F27DA FD3BE93F F88E3D28 1A0E376A 55222888 D05DA2FB 79A31457 690874E5
 9E619740 821CD9AC 21F1D71C A0BB1D64 9CEC3EA2 E587F2BF 99E2B8F6 1311CF9F
 2AE033C1 BC3416DB FF2C2EC6 92BFB870 BCB89126 7101F018 D61E167B A644DF25
 425D17DD 7B7005E5 D5D4735B 5D5AA4AC A190150A 7DFB8687 AFB229D5 3FF98468
 38C3B29F 3144AD67 EE0A1ED8 EABD3F2B 1B41B03E 64871946 44FCA932 25DA5312
 FF5972BB 5054DDB2 74E96E56 E2593F3D F440D528 68B2F45A 7E812A31 053035E1
 5959E583 1C9CA236 1D370850 775E2671 858852CD 7E88FA76 2C2471B7 DB30AC27
 C8D3E2E6 A59C2F5E 4881263D 993962C2 599305E6 8DD106F6 F005F6AC 40174DE0
 0274B4FC 8B3D00F9 864D6434 36045EDC 4512A208 F1F5A603 309924E7 0387E168
 A0295A19 280C1D29 4681E35B 218B50E0 78B67D84 906C143D A69D35A4 1F4254A9
 33506472 35051018 BDFB6E31 8C0620A4 1AC0C0D8 476F3DC7 FA901DAF E0D92427
 69920899 D653D7DC 700B0584 FD8DAC53 32485439 B94184DB 248DDE44 0868ED9D

 $c_h =$

434247A4 4663A599 08F1635F 4E292238 2A234CC0 735FB2EE C013EE8F BEFCC643

 $n_T =$

8B46B532 D32CA5B7 1774B0D2 006A4217 513E6697 02146A31 A83D7585 0642E9E7

 $c =$

A2F2CA51 EDE93FF9 188AB125 D9B82C02 02C4C3CC B0DC8905 CDE1FA30 95B985F4

 $s_f =$

9EC76930 DF5B05E7 59408FE6 A691F8F5 DAE31E1E A2340CF2 0F49982F FECFCC14

 $s_x =$

85953B61 5415407E F7CCF6DE F784EC60 111309DD A99C776C 5BBB77AF 4FBE23D2

 $s_a =$

3A4E1602 1F574809 063E2FDE BEEC154A D7FA13ED 0C4C95C7 86DD27E4 B58339F2

 $s_b =$

48B5E17D A8D835F8 DFE64023 93009BA9 0BA25D4B B5FEE9C2 F7354BD8 2AC3940E

签名过程: $J =$

088AF78B 3EF3F55C C0CB3D6E 29E7DA58 2EA35EFF 8DAC4556 45509925 73E40926
 3AC4611D 126147BE 06632127 05B7196D BEC0DFB9 4C2363FE 784D2236 C8D7064D

 $K =$

B3187052 CB49EA57 AD88018B 2D763FAB C4FAF474 9EEDD937 8D351D5A BCB69A61
 2BA4EA00 2F09B0AC 18943F24 A8DDD28A D9B6652C 4EC77F4D 605BFD7C 056BCAAE

 $J' =$

2E6A75DF 72B54FEC 6F8EC765 D32CF139 80720BF6 E39418B1 0A69AA30 BB983C1A
 82E892A7 8AE32B6C 7C9446E0 FD02DB98 9165E830 9C8438A6 5DC77A34 6279C95D

 $K' =$

F70B4979 3D37BCF8 88B9BC33 0C44037E 1FF02AF3 F660A110 ECC0FD99 823B5C70
 A805C2C2 72C34FED 7586AA0E A06B6A2C 2A3DFE9B C97CF6F1 F1418AD4 7CFC1C21

Message $m =$

0460EAEF 6D8324EC 4D04EC06 42B1EE33 BDC66299 9C39453A 6CE7AB32 CA00EAB1
ED60AA91 4125EBDF 3846C71C 26EE64F7 D49BFAF0 510F2131 87728A83 C639BFDC

$u =$

C2696A55 90152F1A F0B097BE 5BA38764 95D18C44 E34C96E6 DE6AF6D4 E6D6C845

$v =$

A1ED15C0 83ED601C B641CEA8 60940FC7 6C428B7B 5921A76F 9CC8784A 6ABE14D3

$T =$

A65D5F0E 896F767F 386CC5BC 1F5E24A2 C5390881 72768A82 A7606B71 A03EFF28
432237D2 4FCA8C2A 90B6AE11 DA512AC2 61907813 A65F4987 C5E66275 FADF6C8C

$r_u =$

4123691F 44DDAF84 B07B74B3 79355352 D31BF942 B967DE90 E8E7407C A49A0BD9

$r_v =$

B3082FAF E4E27122 3CDB2148 9B5424AD CD393739 1C55ADD7 4E637C6E 680EEBFB

$R_1 =$

9A4EB3A1 5AED2CBF 8C706159 58C2936B 400A81AB CD53C007 399BFFF7 3429ABF6
E5B2AF58 D2F70AD7 8C551940 50FB7971 5FB607EB C919703D 55C9C782 33285D25

$R_3 =$

2E34680B EDE498D6 6BE819A7 FEE2877F 0D296C9D 829590BA 472604DB EC2E1F6A
1EA38AEF C0BB61C0 91151DF5 1953705D F5BE1131 02744B90 E4955BFD D8BC13C5

$c =$

5FC97235 909666BC 21EDDEE1 A7BFA1D2 A965D936 63B49BC8 FA456F1C B150028E

$s_u =$

7031B184 779CCFA1 FCCB306C 5E001530 003E2299 CC26AA85 C5D949A3 742E2AB2

$s_v =$

DE394CFC 2325071E A86D33FD D7314643 2E6069E5 D4D6742C 111C448C E37A19D6

$R_1 =$

9A4EB3A1 5AED2CBF 8C706159 58C2936B 400A81AB CD53C007 399BFFF7 3429ABF6
E5B2AF58 D2F70AD7 8C551940 50FB7971 5FB607EB C919703D 55C9C782 33285D25

$R_3 =$

2E34680B EDE498D6 6BE819A7 FEE2877F 0D296C9D 829590BA 472604DB EC2E1F6A
1EA38AEF C0BB61C0 91151DF5 1953705D F5BE1131 02744B90 E4955BFD D8BC13C5

E.4 机制 4

安全参数：

$t = 256$ 。

群组公共参数：

G_1, G_2, G_T 是由在 ISO/IEC 15946-5 附录 C 中 C.3 规定的 BN 曲线构建的。设 q 是一个大素数。 G_1 是椭圆曲线 $E/F(q)$ ，其中 $E: y^2 = x^3 + b$ 。 p 是曲线 $E/F(q)$ 的阶。 $h = 1$ 是 $E/F(q)$ 的代数余子式。设 $k = 12$ 是 $E/F(q)$ 的嵌入次数。 G_2 是六次曲线 $E'/F(q^2)$ ，其中 $E': y^2 = x^3 + b/\xi$ 。 $E'/F(q^2)$ 上点的总数为 $p(2q - p)$ 。 $E'/F(q^2)$ 的余子数为 $2q - p$ 。 G_T 为 $F(q^{12})$ ，其中 $F(q^{12}) = F(q^6)[w]/(w^2 - v)$ ， $F(q^6) = F(q^2)[v]/(v^3 - \xi)$ 和 $F(q^2) = F(q)[u]/(u^2 - \beta)$ 。

在下列数据示例, $P=(P \cdot x, P \cdot y)$ 在 G_1 中表示为 $P \cdot x \parallel P \cdot y$, 其中 $P \cdot x$ 和 $P \cdot y$ 是 $F(q)$ 中的元素。 $r=(r_1, r_2)$ 在 $F(q^2)$ 中表示为 $r_1 \parallel r_2$, 其中 r_1 和 r_2 是 $F(q^2)$ 中的元素。 $s=(s_1, s_2, s_3)$ 在 $F(q^6)$ 中表示为 $s_1 \parallel s_2 \parallel s_3$, 其中 s_1, s_2 和 s_3 是 $F(q^2)$ 中的元素。 $t=(t_1, t_2)$ 在 $F(q^{12})$ 中表示为 $t_1 \parallel t_2$, 其中 t_1 和 t_2 是 $F(q^6)$ 中的元素。 $Q=(Q \cdot x, Q \cdot y)$ 在 G_2 中表示为 $Q \cdot x \parallel Q \cdot y$, 其中 $Q \cdot x$ 和 $Q \cdot y$ 是 $F(q^2)$ 中的元素。

$p =$
FFFFFFFF FFFCF0CD 46E5F25E EE71A49E 0CDC65FB 1299921A F62D536C D10B500D
 $q =$
FFFFFFFF FFFCF0CD 46E5F25E EE71A49F 0CDC65FB 12980A82 D3292DDB AED33013
 $b =$
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000003
 $\beta =$
FFFFFFFF FFFCF0CD 46E5F25E EE71A49F 0CDC65FB 12980A82 D3292DDB AED33012
 $\xi =$
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000002
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001
 $P_1 =$
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000002
 $P_2 =$
E20171C5 4AA3DA05 21670413 743CCF22 D25D5268 3D32470E F6021343 BF282394
592D1EF6 53A85A80 46CCDC25 4FBB5656 43433BF6 289653E2 7DF7B212 BAA189BE
AE60A4E7 51FFD350 C621E703 312826BD 55E8B59A 4D916838 414DB822 DD2335AE
1AB442F9 89AFE5AD F80274F8 7645E253 2CDC6181 9093D613 2C90FE89 51B92421

SHA-512 作为基础的密码杂凑函数。实现特殊的密码杂凑函数使用附录 B 中的基础构件。使用优选的 Ate 对作为基本的双线性映射函数。

群组公钥：
 $X =$
81ECB895 667EA4F9 F37193F1 EE91968D 0E1677D8 42C9D98C 0731486 D1797A492
0F31D669 D93543F9 23484F76 3EB07485 EAD88D90 EB277476 7F4A599 00253F849
FF83F12E 98791CA7 63A900A8 94CF2690 6E42CAB4 E96B614D 2E2F468 1B7B5D1B1
BC97D3BD F100EC4B 16635FA0 3B4959B5 58ADEF4D BE6D8904 0CFC739 9A294195F
 $Y =$
A7F6DBE3 D5FE924C 92B87B9C 87D25132 FB464A8B 48032A70 DFD4844 B588FE585
504147A8 64F90C5C B22C49D3 2B9357CA 51760D52 621CB632 50D522E AAB9BB271
0910BEEA 0B55068B EAE74888 75A02E51 46B37C9C DEC6B2B7 C74FCA2 9E2ED2AAB
4E148283 F3E99483 8A24F2C6 903EE6BD E99EEFED F2D137F6 3BDED47 BE46297A8
群组成员发布密钥：
 $x =$
65A9BF91 AC883237 9FF04DD2 C6DEF16D 48A56BE2 44F6E192 74E9788 1A776543C
 $y =$
126F7425 8BB0CECA 2AE7522C 51825F98 0549EC1E F24F81D1 89D17E38 F1773B56

群组成员发布过程：

$n_1 =$

D2815256 86B5897A 1396AA5D E0509543 E319E3EF 3EC1BAB2 24CDE6E9 5BE18CDA

$f =$

05E8D2E3 F942A58F 652CE4B7 2836BB01 23AF440F E74004CC 0E0F37F5 59BAC367

$Q_2 =$

2F858C21 7C1F2818 F1912A72 20852462 8AE6FC53 49A97D82 D6ACB646 AD3A4284

B1A886C3 3E5443AF 1499EF32 F0CB5186 B7F25E52 FBA05426 CFD590B1 974143DF

$u =$

84A7A234 A62153E1 158405C5 C8A64EC6 6ABA6220 CA230421 460C3F4C ABF83879

$U =$

1B740B6C 6D316705 8193305C 5D9E744D EC93493A A9A28539 60410205 590E7CC2

49038FC8 4B9BBC70 39330E1D 6A01F156 8D537112 CFB566BC 5EB6F470 8F83134F

$v =$

D4338FA7 9437AC22 04DE4799 4D0D728B 9745DB69 74FAF25E AC7409B3 8E562067

$w =$

E37A4E0C FAF2ECDD C3F99A21 D7D293D4 C3E30982 AF0DD17B 0E6494E0 5F4721AD

$U' =$

1B740B6C 6D316705 8193305C 5D9E744D EC93493A A9A28539 60410205 590E7CC2

49038FC8 4B9BBC70 39330E1D 6A01F156 8D537112 CFB566BC 5EB6F470 8F83134F

$v' =$

D4338FA7 9437AC22 04DE4799 4D0D728B 9745DB69 74FAF25E AC7409B3 8E562067

$r =$

7FD81AC7 9FB82ED5 003ACD4C 0FB73365 CA30CABA 724AA5FB 4FBBBDBB A708F736

$A =$

8280EA5B 44404EA4 1468AE3A 3273793C EA5A03B6 88DDFA1E 14E47244 2B45ED8C

49DF9DD8 350C7294 AC9AF0B0 8608969D 4CA0CE4A F62C7A23 E87BF770 3EBDAD8E

$B =$

DE8C62EA 37A647AF 3450AB3D 88219603 80A0700C D4BEFF66 E7B33986 D4399D15

FED6B937 580468CD CF862F71 F12936E3 261EA0C4 0FE2C24A 5A50BC51 9D94D039

$C =$

6A8DF72A 3D2E7AF9 C8D68FFB EA8174A7 BA44A04C 39D2E72E 6A8ED491 390E5314

8DBEB136 02EEB668 11E75FE0 DC9F856D FAF990C6 6B10778A C23078CC 19C7FA1E

$D =$

B4C3017D FA57BA15 7B738B57 3C52F553 70AAC3A5 3FCE4132 4C13DDBF 2F54DF96

08B03522 315970C5 A9493A07 7C57C6EE B11AB863 834554E2 6DA1B10D 87053A8B

签名过程：

$bsn = \text{NULL}$

$n_v =$

F7BACAF1 4B704BCE C0CFC1DA C078EA18 94D2FEE8 ECD79D96 472588E0 EF1DD3FA

消息 $m =$

BD06E6EB D43C94EC C8532643 93825074 FE1BA5F4 E59DE3F8 38E52146 F90C925F

EF39BA7D 98AAF8F 07426D7C 3F5E39D8 1B7B358D EB330205 71C311D1 494E430C

$J =$

525998AA 0F1375C5 675CD264 66B2191A 6CDF0ADD 4B8D815B 27621253 1BC12924
4CB95CD2 AFC76DF1 31BE43D3 DF81866E 0E90BFD2 816E8D00 C434BE4E 4A41AB0D

$l =$

D954D9B8 C3168F2D 926D925F 8675FB1C 34BA0C67 45D472C7 0907571B 0E6460C3

$R =$

83AF9C12 8B18FF37 CA710FBA 5B67DE20 10786545 A275AE4B 26BE10F8 F83A3EE1
624093B2 67186E40 FFCF788E EE7F538A 9A9FBE61 F7E74ABC 6B6FBA20 CCD71FA8

$S =$

6B72A84C 752AE794 0470B90A 497D4EEB DF931490 9061635C F26B3D57 69D62B81
859DB022 EBDC2CB2 D9EA337C 9879E647 83ED4EE7 0D5122CB 45B869D8 6A078112

$T =$

9E02290E 7681DFB3 2D6DACFF 7A7EF05A EA6D4915 A260BD0D 44922123 BB844524
411E058A 1184E12C C97BF7D3 A51D3FD1 94553E54 B0DCB4E0 E11B17BB 7E7F2457

$W =$

2D476F4F A32726E6 55CE2338 AF6D4C74 FB53CBFB E70D3089 4CB9D51A A435B836
86EB2076 AA7EDAB8 22AA7AFC FB87A21E F34B4266 B83A1A93 CD405BA3 EE3A4A5B

$c =$

C0B08FEC 2BEE0485 DF041E36 8510FB80 0388AE8C 6CF582F2 7CB36FC1 F69FA594

$K =$

C8C20A5B 4556E0B3 02224900 4BCBF8CD 7B0F6D0E D06801E1 E4835E1F C3226538
60EC31A5 D75EE611 04EB3A11 4C6D58B8 B7532E26 4C2F58D2 5B43C479 6979F7C2

$n_T =$

38C2842F F986CDF6 BE7331F7 7A8FFCD0 7810CC43 CC07CA27 7B33B109 7D3B685D

$r =$

DDC650F8 29907BA5 1B255597 BF83ECE2 183A8D6E 3A2AD054 DBA6A7D9 7AC5AEE8

$R_1 =$

E57877BE F012310D 15C5EDD6 32B747E4 DB9FFD06 A3539485 0419DC2B 8B456093
D80DA3FB DD5A4EE6 2F80C4D2 5CD69407 477C350E C64022D3 195FDEA3 38B6A1D1

$R_2 =$

ED9A846A 104EB7C5 0361C67B 2FB92A05 8CD245C7 ED9F172B EB449F41 3D675CD4
86B17A09 A511948D 09F47D96 2D36E4ED D8BEFF84 0BBE9515 8D55C98F C65716C8

$h =$

B329C912 53094139 4DFC1162 043A458A D9CC21C8 4F08F22C 6BB480B5 362D8F89

$s =$

E3D906C7 D2D5D0B4 BC13ED90 DFDC26B4 AD86C66E C9793898 F54A3305 0BE591D6

验证过程:

$R_1 =$

E57877BE F012310D 15C5EDD6 32B747E4 DB9FFD06 A3539485 0419DC2B 8B456093
D80DA3FB DD5A4EE6 2F80C4D2 5CD69407 477C350E C64022D3 195FDEA3 38B6A1D1

$R_2 =$

ED9A846A 104EB7C5 0361C67B 2FB92A05 8CD245C7 ED9F172B EB449F41 3D675CD4
86B17A09 A511948D 09F47D96 2D36E4ED D8BEFF84 0BBE9515 8D55C98F C65716C8

$h =$

B329C912 53094139 4DFC1162 043A458A D9CC21C8 4F08F22C 6BB480B5 362D8F89

E.5 机制 5

安全参数：

$K_n = 1\ 024$

$K = 160$

$K_c = 160$

$K_s = 60$

$K_e = 504$

$K_e' = 60$

$p_1 :$

C4728B13 4D6367AC F5237597 0C8F6C6F 3FC8F2C0 15225558 77F0875B D8F52547
B29C45F3 8497CD20 0FCA69FA 3B51137D 270D6E43 B9FB4D9E 7149C7A8 9E6FC52B

$p_2 :$

F5912C12 576943C1 2182E9BA 721AF44D 3EB24401 19A4787E D62EC771 8D53D179
72DAF22F 0B8EA3E4 659BC08A F3C1EB66 3D813830 68AE9DAC 2B0DAB68 9EBECBCB

$n :$

BC7105CC 0207C46B 8F10732D 97BF0C2F E410F50C 6B79AEA7 68AB94EA A67DA9DA
65FF0101 1DB263CA 530E74ED A590E7DD 7CD015A5 455CE962 FD477340 B9014C29
FDED582A 9535C3C6 0039D828 86D755A5 CEF4687B C15C8F85 06F2A083 B7B5E59D
F91AFFD3 371F6879 0F43B22A 67DBC6AC 8DE5D083 65183679 DA0D6D75 23E47219

$a_0 :$

10A4AA61 0DF057A6 40E42291 0FA165EA 46774220 26CCD4D9 A19F7FF3 FC8E4E61
14E34C3C BD72385E DA6010D7 4CC0D370 C0BCC71E 69CEBE97 7BBFC07E 57E68CE6
42223E3E A14BD7EE EF09ADC7 BB2112D6 B905540C C25AA65B A78EB30B F27B702D
23D62F56 FB51D688 07A2B866 4C78FDB7 60E4A5D7 377DE475 C7DC385F 9961BB1F

$a_1 :$

9AF65DE8 B26D27C3 DABB8667 87797522 9D6B52EC CA376B60 44D4737E 0C0CE9C4
14B466E2 226204D1 4658AED3 82DBF8BF ABD7C69A 90E947A6 9E6B165F 102EBD57
B5E04D8D 5C3A55BE 2C40929A BCEC6637 00777B8E 561CE99F A894F270 7F6D02A2
A0BDC0A9 D19D2812 56923C10 2A20D37A 1FA9CDE2 EC253DAB A9093CFB 5BEB9236

$a_2 :$

92EB72CC 73B6FABA AD963859 5821FF57 574E9002 B6094441 9E1F4BCF 2BC5568B
18BAE51B 888FD457 28A7F369 4A0E9822 7A314374 9ECF5C88 8AAC9A99 2311DFF6
2EA1E5DF FAE0BF72 DD7BFA40 D8FA28CF 5BB220F5 F5052C00 319335E1 DBE30A23
30418B47 0DCF2F83 A735B6CA 95E6BAD9 827E67AC 92289E9C BEF4E77E A3B682EE

$b :$

85B97490 2BA79FE1 1005CFC5 1E31F655 5434A59E 02A34BDE 07BC586A 1B815660
E421DA66 6626CE82 81158F20 3A781612 85F70C44 802DF4F5 322C2E13 5892B0B6
2968296D 2856B268 54BE8A86 54D30912 218F70B5 D05E3ADB 26D551B0 B5CCDA13
25A9186C 8693FA6A 6753C7CC 12E3C723 AF6086B6 136E3BA3 059B4620 D8326111

w :

7429D36F 39BB19A9 D70AB651 23918060 50FF943C 3B571E40 A4951DCC 360D757A
C328D524 A49D7A48 05A93F31 6CF04065 778A6C00 6609B17C 717BBDA8 9852EF05
84406761 DBF91455 E519D589 EDF0D30F 92E23D53 05430F9B 12479268 97335813
A6A13D98 3431F26E C7EBB0E5 61F3F106 D558E18E D5A81D44 97439911 4DCF4DC5

q :B32DF688 513664AE 1140F824 301CE778 217C9AAD

群组 G :椭圆曲线 $Y^2 = X^3 + ax + b \pmod{p}$

p :B32DF688 513664AE 1141034A E8E96559 1DA38F43

a :A559B0E0 588FEDF7 52A8E990 66E6F28F A2A81926

b :7C1EA8D2 84973DDF 71E4782C A06ABF7D 8535B163

$g = (g_x, g_y)$

g_x :A0F22B79 45A70BE2 64F5BF5D 05FA44BC C924A6F5

g_y :6055AD8D 1A371676 65457EF6 F7AF00B2 31C58D61

y_1 :8209EBE1 69FDF9EE 42DC00EC 8F80401D 4B6B9763

y_2 :AED41C99 D021A81E BA13626C 269A310E 84F34E6C

$Y_1 = (Y_{1x}, Y_{1y})$

Y_{1x} :055888FD 19A5E1CC 6E711218 478DA01B 3CC74267

Y_{1y} :843D8699 56281896 60B7FCA5 6CD9EB55 BEDE13F3

$Y_2 = (Y_{2x}, Y_{2y})$

Y_{2x} :6855CF69 7DC43B51 F9C0CA54 6F81F50D D3F9E115

Y_{2y} :58D96EBD 67B43408 AB860940 9B2BDFC5 AD379BCF

x_i :

096B6D8A B16450DD 322640DE E30806AB A72380B8 9F88875A 92976AF3 FE63CAEF
00866DAE 0AEDB5E6 92E5E6FC 1FA9E0BE 9B4B71E0 0B637BC0 56F68366 112E7D74
84FD8515 C0C5A760 BC7765F2 ED44159F 82A2163B 2089F17A EDBC342D 94CB4715
2AF9CDD4 61AE71A0 94D455A7 01D86DA4 E70681F7 97545EFB B132EA63 D743A6A6
9C1F0066 53A26292 FE950B20 1BA26A54 C2159AE8 332A13EF BD5CB73A

A_i :

BE3A25 96430AAA B48EA819 94A54E5F E150F9D0 E14AE868 FC93D743 9914B853
F712CCC4 1303554D 81D9028B 9F7726BB 2EEDDC2A 48BC80D5 B1DED4ED 1DFA9847
96FFF199 47226633 745152C6 6B342261 7D1CFDB9 ED66A52F 43ADFB4A 49C42900
64F5708A F86DD9F0 715AD701 EC2DB64F D9510F76 084FA8DE D39529A0 4EE119EC

e_i' :017ECCE7 23FF93F5

B_i :

4691B234 C873C9BE 98568307 87603355 71463B9D 6FE70E2B ADE6C3AF FF2E5DBA
31CB568C 5EDDE398 F5629F3B A6FD1617 3BCE054C 2147F932 010FD305 45A81600
AF5D46FC 4DABCD9B C90BDB5E 1A1E6187 B3B6CDF6 CF401C41 73648A6D DD898BA5
87792BE0 29D19E77 F2B75899 95B2880B 1CC1B548 284ADAFB 63499DEF 5907278B

$h_i = (h_{ix}, h_{iy})$

h_{ix} :4518BDDF ECFCD0E1 1D773410 E364708B CEF86DF

h_{iy} :8AC076DF 5F83F0B4 AA071A2E 06885254 BD97E8CE

x_i' :

061EEA6B 26FAA61B C1CA23DF 36A2A670 0D864916 9D0A3765 CCFC8945 86E701D5

374B9973 FF171860 3EB23114 9B048205 71FC351D EC6E9DA6 92B88DFA 24967AEB
 38D7CBE3 2F7CB5EE 2F56665D 74598717 51C4F7BA 393DBE0F 19AD70CE 1A8954DC
 ABED6AEB 2CF0A650 DDDD37DD E7FA29EA 619B4619 CC53EC42 CD624B9E A7246BDC
 EF7566ED 5CC75E91 85A16F60 A88F6810 8AFFBE21 0981D34C 249D8A31

x_i'' :

034C831F 8A69AAC1 705C1CFF AC65603B 999D37A2 027E4FF4 C59AE1AE 777CC919
 C93AD43A 0BD69D86 5433B5E7 84A55EB9 294F3CC2 1EF4DE19 C43DF56B EC980289
 4C25B932 9148F172 8D20FF95 78EA8E88 30DD1E80 E74C336B D40EC35F 7A41F238
 7F0C62E9 34BDCB4F B6F71DC9 19DE43BA 856B3BDD CB0072B8 E3D09EC5 301F3AC9
 ACA99978 F6DB0401 78F39BBF 73130244 3715DCC7 29A840A3 98BF2D09

C :

6E208796 89C180B6 7B9AE0AC 5116FDC8 300CDFEA 0E437793 0FB53C95 60736431
 DDB768FB 235F613F 76A44259 9C960482 AA8ED381 AE3C5D08 B087DD17 231FC9A0
 FAEC0736 CF0D7657 A0413EC2 F36FA9E8 9836BA2C 7C12BAFE 5913ED1B 9434CD81
 5CF2DE17 7ADD702C 9894C0C1 F66C41D4 4E790879 B5F259BE 3A8383FE 90BC4CB1

A_i' :

91C44789 253105A7 D14B7123 4C73FB86 6B3522F5 C543D9C0 0A83EAA5 35C81E1A
 68B98CF0 31D77CF3 572930A2 F9F85C42 5AB673C8 F7B6F9C4 9E97D169 C01C8702
 9473DC31 A1A76449 0DB11DB2 C083B45D 7CA19AED AE0DBE93 B7EEF152 B8517C7D
 E8FD2078 CCEB2044 6C5AC4F4 456C8114 01EB03B7 7094D79A E37467FF CC04318E

签名消息:

“*abcdefgh*”

$E_0 = (E_{0x}, E_{0y})$

E_{0x} : 4057644E 063743F4 F3A12C15 93F745D3 D7383C1B

E_{0y} : 7EF24989 EFEA4D2C 0B34C752 028B8C5A 07735E91

$E_1 = (E_{1x}, E_{1y})$

E_{1x} : 9C4F01B8 293BA945 DF70435F 9FF4AF50 B15A52AB

E_{1y} : 60970938 977FF31F CF0724AB 546C0856 F94F4C3F

$E_2 = (E_{2x}, E_{2y})$

E_{2x} : B1D29A8F 3047EA63 FF45D35A F7C6D3C1 5CC512EF

E_{2y} : 440A9E9D D775A1E7 FA4E0B07 E6F2E10F 18A1AE92

A_{COM} :

3C9E3686 BD8A737B 18F8C001 DB447140 932C22D2 CFF16A07 436B0C9C
 D7A4ADCC 359339DC F9161C85 533ECCD1 40A33619 09B30FD1 9F57EE86
 983D168F 7DD92571 9E91327E 64885FBD 65922911 18DAE05B 51E79242
 A6C899F6 61619119 FAB1B282 333744CE 0E9206CD 81E5BB1C 488B349C
 49179328 F22E1FAF 1B07ED1C 02C18B6C

B_{COM} :

3252A9E8 B3E9BAC5 22FDD692 03350BB5 BCECBA93 1181B5CE FD7F7395
 A79E0C07 22033899 A539D3E3 C38DA3BD A85B2AA6 9A91C43C 592C83C2
 4B63394B 8C62CCDA D9C8B15B 75B9F123 7291FAC5 8CAF3A1F DFB1DC06
 FD1DC3F4 183AC1F4 E53A4F74 A488ECD3 C6ADBBAA 6B78676E 1B6B103B
 B28AA33E C9DA0A82 E095599D 5AD93AD1

c : 8A39EB36 D25CF1CC 13921BB7 294556F2 0B070E91

注：为了计算杂凑值 c ，每一个部分的比特长度需要补充插入。

τ_x ：

91D961 63B56FB9 C1F07D1F 060B7A2B 40AEAFD0 05A0DA6D B598A751 9252D68D
 896E5B0F AFEBCF66 90FAA2A0 4D8FAC62 EF07F707 6D3E12CA 4A3F1556 D521B3DA
 103F5F69 FA0764B0 7287F5CB 5A33A7BA CEA6DF78 0E95C72B F05171AE F750AF48
 48788F47 7F692386 B2BCDE54 85D7DF5B 5A8A8978 6189ED17 CC9DE013 FB827836
 C0BBE630 42A05DA3 27B1D03E 061064F9 42066EDF 213AFBDC B52A8E98 D787806F
 2549E862 7BF60435 A81EABC0 5B0E55B4 2411C28B AA9B2F0D

τ_s ：

05487C E6BCA419 064BDECB 349C1360 A62FB37B C0A300F3 19401FBB C60B12D9
 286D55E7 D3E38B8E DD92B306 41608985 87D561B1 912A7BC9 54116E15 FE7FC0C3
 8487ECCD 7AA342AE 87233A90 01620B02 51FCBCFC CBDD2F7E 48ECC809 B3528EF6
 D3A3B478 873D0834 29744CF7 9031B0A2 8F543E3D B527F5B3 DE2AA3A9 28FEB5D6
 1CAD83DD ED439C88 159DD1D4 0E155923 B66EBAE9 4FE4EDF1 3175E2CE

τ_t ：

830541 B18BE4EE D1FBE313 D4241E01 2A17F3B8 399867F9 5487A766 57E9E7B9
 ADF2E2DB ADA69FB9 A93607B5 72E0EEE7 B4DE654B 3858F894 30A5ABF7 C65B9E2A
 056F5239 7AE35EFD 72A25B09 1D9CA64E AB3B9E1A EB219CB8 FE382136 E64EB78F
 1EA503C2

τ_e ：

487773 182405D6 0191A1C4 11C412E1 B6916B5C 65D8A9F6 B39FA82A 8EEC37C0
 B3EDB5FB

τ_E : A81030E5 81B57BAC 6844588F D9860A03 1145DB12

签名过程：

ρ_E : 18287414 95717964 B262DF85 86FCC4D1 81D07793

ρ_m ：

81696D94 9AA69900 0C04F70F 19DEB5AC FE38B1CC 81FABC06 6CF39591 9B397CA4
 B995F2F6 AB98376F 291847B1 5941783D 291FF6D4 B72B8168 188EC101 D352555F

s ：

40B4B6 CA4D534C 8006027B 878CEF5A D67F1C58 E640FD5E 033679CA C8CD9CBE
 525CCAF9 7B55CC1B B7948C23 D8ACA0BC 1E948FFB 6A5B95C0 B4CDCA46 E04F56E2
 27CF0EEA 89DFAC3C 6C117744 248A8A93 9380B3E9 247B14A7 E642E510 BE0A1740
 4E95D316 738756D1 0F6AE84A 0B31158E B5F6234C 55548B35 A93D407F 8B7240EB

e_i ：

800000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
 00000000 00000000 00000000 00000000 00000000 00000000 017ECCE7 23FF93F5

ρ_r ：

E4946C9F B4DEFCD8 236ED484 92269CDC 5C1F16B2 00A21DA9 403B4F4F AD8C3053
 349EF229 3DE30F5A C3D9F0D9 F3E9F4D0 9A44A5F3 0BF66F5D 80E52E06 1E59B238

t ：

0155CC6E BEF1ABC9 36729308 F4E9075B 13CAF958 2EDEEFE3 F9A735A9 F32AB029
 D8901F58 97ECE5A0 2D458E75 1B1EAC7D 20CBEFC2 B03766DC F05C533C 4828B0A0

6F857E86 BAF5B798

 μ_x :

91D961 63B56FB9 BCDA727F 3C94EFC8 B9328B4F 972C4F6B A7E927C7 AA0BD9D9
 79EE456C 0B4F05FE 321FA976 4FF6F98D DA54D2F2 55834955 5D71267F 701E44C4
 F03FD80F 832FD1D9 51846426 3E27EBEF 4594B39F 7B43EC59 DAD20EE1 8593F7A3
 3C1622DF 66140AF8 18CE1D5D 2BA64ACC 0DA0EBC1 4DCB25E1 EDC06E69 E884D20D
 E073182D 57B9DA9D 4DAB48C0 C290FD77 BE9428E0 F06B6128 07099423 0F3B4F40
 F5184DD7 9C385B85 1F3B949D D0B35257 2F9F8129 D67C3B33

 μ_s :

05487C E6BCA419 0628EEBC FB1C4342 3403FE93 C259969F 0E7D5DC3 408FE40E
 407BA627 9B9C40CD 1AB4E015 8E7647B7 749C40FB 68F47643 3DD76642 FB49F566
 92A92D35 44C68763 8CB9FF00 4FC40108 FE21626C FFC7BDCF 8E4C06D2 3396EA77
 FF9A6E3C 148421CC 3E3A68AC F44E601B 4CA3AD9D D27190C9 FC00C83D 9CD5901C
 5930B83F 927ABA37 DA15BE2A A170A366 135ECB38 860A6303 16C543B3

 μ_e' :

487773 182405D6 00C2F0B4 20D35359 AFA292B2 80DC0D76 30C3947F 3CB061EA
 C3608236

 μ_t :

830541 B18BE4EE D143558B 90B1AD28 2CE8F67F B8B865D2 C5E3F969 5BE60AAE
 B286958E 54A4C8E9 CB43FCB5 879529FD ADFCEF8C CD318E14 BFE906AE 9BDCD11B
 3C52CA4A 3A791E3E ED544119 E7AE1E4E 57D777EC BFE13BA4 ACADA4B3 132265BA
 3C45B6AA

 μ_E : 4E52BAB4 1AF5F83E F503F0AF C0207A30 88650CA1 $V_{\text{ComCipher}} = (V_{\text{ComCipher0}}, V_{\text{ComCipher1}}, V_{\text{ComCipher2}})$ $V_{\text{ComCipher0}} = (V_{\text{ComCipher0x}}, V_{\text{ComCipher0y}})$ $V_{\text{ComCipher0x}} : 50196E34 \text{ AEE613F5 DF1CBDFF ADF6ECF6 2253BD3D}$ $V_{\text{ComCipher0y}} : \text{A8520B66 3AFC36CC AEE39596 7D9666B2 EDB8B35E}$ $V_{\text{ComCipher1}} = (V_{\text{ComCipher1x}}, V_{\text{ComCipher1y}})$ $V_{\text{ComCipher1x}} : 62024C0D \text{ CC109670 F73110FD BFCBF5F6 CAD86358}$ $V_{\text{ComCipher1y}} : 986E2628 \text{ 54B6178B 46516F9A B068D686 F447FFC6}$ $V_{\text{ComCipher2}} = (V_{\text{ComCipher2x}}, V_{\text{ComCipher2y}})$ $V_{\text{ComCipher2x}} : 8261A8D1 \text{ C41CAFD6 94299D0A F2C7066B 8D8372E3}$ $V_{\text{ComCipher2y}} : 8B3BE9FF \text{ D55E06E5 C95EB9B7 3DCAFD61 35E5A274}$ $V_{\text{ComMPK}} : 348B4F92 \text{ 545D34E9 7057B93D C737F8D7 E418D7CF 62AADF17 22DCEF15}$ $A1F7C2C2 \text{ EEBAD921 AD388D1B E12EB457 BFCD88A8 5CA7EE6F 2A2B5C46}$ $57F0BAF9 \text{ 634599B9 8087B1D1 3E18F691 0F2CD14E 514853CA E9435EED}$ $421EB2DC \text{ 078BE2D4 A2046315 166780FC 42E3B4B2 2F71E897 B480D859}$ $3956C923 \text{ D02DB006 41BE61D7 8B4A1C14}$ $V_{\text{ComRev}} :$ $2B9C2D39 \text{ 4A17ABD3 00DD0034 6F3C2595 14985E73 708B709B A07E6BBB}$ $E0752C39 \text{ 3F8D3079 4F66FB06 2D3E87F1 3B4F1C95 0CBBB72A FD715EB1}$ $E153D2AC \text{ B985EBEC 163D308E 6E83B499 CA648183 61BED12B 0325DAD0}$ $3C2B4B79 \text{ 90AE8B6A 3663BDDF 19D86F31 ABB1F9BF EE091B21 F84F7A9B}$

2CAC285F AF4CE390 7B626F29 AB5E7F68

在该示例中,使用 SHA-1 作为密码杂凑函数。

E.6 机制 6

建立过程:

大素数 p : 28B1 A9F770AC BE784572 2E6F07D2 EE10CD77 0D773261

群组 G_1 : 椭圆曲线 $Y^2 = X^3 + aX + b \pmod{q}$

q : 0D5A4B C530F8AE 7F76C977 4304FD7A 5F8653CC 0A221045

a : 010B23 6060B834 1698AB96 34C0A78D 64A97EC8 505896F1

b : 0B4405 046F8846 5245724A D983AE5F 9633563B 6970E263

生成元 $g_1 = (g_{1x}, g_{1y})$

g_{1x} : 005A37 AED5FB1B 415D7CFA F425FC67 FE0BEBF6 CB54E228

g_{1y} : 0B65C5 D3B9ABB4 9BD1E493 D6A8CD80 4E776A24 95DAED81

群组 G_2 : 椭圆曲线在 $F(q^6)$ 上的 $Y^2 = X^3 + aX + b$

不可约多项式 $f(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$

q : 0D5A4B C530F8AE 7F76C977 4304FD7A 5F8653CC 0A221045

$a = (a_0, a_1, a_2, a_3, a_4, a_5)$

a_0 : 010B23 6060B834 1698AB96 34C0A78D 64A97EC8 505896F1

$a_1 = a_2 = a_3 = a_4 = a_5 = 0$

$b = (b_0, b_1, b_2, b_3, b_4, b_5)$

b_0 : 0B4405 046F8846 5245724A D983AE5F 9633563B 6970E263

$b_1 = b_2 = b_3 = b_4 = b_5 = 0$

生成元 $g_2 = (g_{2x}, g_{2y})$:

$g_{2x} = (x_0, x_1, x_2, x_3, x_4, x_5)$

x_0 : 03AE9F 2C466DF7 D4102FB5 E00E2C24 2461C4B0 C8FBFF38

x_1 : 0A982A D3A5E571 0FD93961 1F05DE85 2459A355 F31C1C07

x_2 : 0457CD 3FA8ED6E 809A6988 22B46038 15F66A8B ECEAAF97

x_3 : 0B49A5 D5AB4FEA 6856835F 0C51782B 7A88F81F 54B303F4

x_4 : 01F69F 3707ABB0 F80420E9 ECC7A536 3935E09E 19D66727

x_5 : 09712D CF9DC21C 13ACD3F7 E888A612 5C733700 21EE2977

$g_{2y} = (y_0, y_1, y_2, y_3, y_4, y_5)$

y_0 : 0AAFB2 AAF3C4B1 08C39C44 BA48D8C5 A691B450 C81BE831

y_1 : 08668A 514EB258 962AFD10 DAD3EE78 0C6BA296 6046488E

y_2 : 02B1AC D28AA543 8016E0FF C4196DD3 E3563594 F2EC3A88

y_3 : 08E6AF 4EA61CB4 6D9FBB15 0C777586 337F8693 F5459C72

y_4 : 09FD0E 86DD3AB7 1F84F25D BC633E85 E2FD5DFB 22DD215A

y_5 : 0C04BE 5ABE0D20 471DA9E5 356A1A8E D98B61BC 46205562

群组 G_T : 阶为 p 的椭圆曲线 $F(q^6)$, 不可约多项式 $f(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$

q : 0D5A4B C530F8AE 7F76C977 4304FD7A 5F8653CC 0A221045

p : 28B1 A9F770AC BE784572 2E6F07D2 EE10CD77 0D773261

生成元 $Y' = (Y'_0, Y'_1, Y'_2, Y'_3, Y'_4, Y'_5)$

Y'_0 : 0703EB 2B435CE4 F05F2F42 24068BA2 740DA7DC AA6261ED

Y'1:086A78 678DD736 5AE16EFD 6A0B7D70 AB9AF018 CED826A6
 Y'2:02F7B1 5A902C35 7AFA683D B9596559 5EB1DBCA DA8D11F2
 Y'3:0B0642 FE184584 0D1A8116 B555B875 BE36603F EF2F3273
 Y'4:090F58 77ECC58F C3D372C6 95C620A6 CBD1741C D78ECD01
 Y'5:095212 30C60F99 063757B3 BBA0E033 51DFE18B 316359CF

群组 G_3 : 椭圆曲线 $Y^2 = X^3 + aX + b \pmod{q}$

q:28B1 A9F770AC BE784572 2E336CDA 00D5021F 4BD1AE6B

a:0635 BCF4552E E155F24C 87D90C32 F452715B 0C3C4C4D

b:1C46 300EC64E FBCC60D9 1E815474 18301F69 335915D1

生成元 $g_3 = (g_3x, g_3y)$

g_3x :1D24 0DDEAE24 90078C20 EA2CB1CD 4EF78348 9B5FD08F

g_3y :0BB3 1A110316 8A38E1B8 E26F1B9D BEBB7D43 B3267066

G_1 的随机生成元 $P1 = (p1x, p1y)$: 和 g_1 一样

G_1 的随机生成元 $Q1 = (q1x, q1y)$:

$q1x$:009AC0 083AF055 C2DF779B 98891B71 DD97EA3E 21FC1CBC

$q1y$:0762B4 53CDEBC2 E80F08EA 1C8CF3FD B3997A7E ADF3BA29

$G1$ 的随机生成元 $R1 = (r1x, r1y)$:

$r1x$:07812D 2823AC87 25A06CBD EB104669 BBF14AC4 A420B945

$r1y$:03464D 69EF68A9 2EEAE848 DBACE27F E378D544 DBA9AE00

G_2 的随机生成元 $P2 = (p2x, p2y)$: 和 g_2 一样

G_3 的随机生成元 $P3 = (p3x, p3y)$: 和 g_3 一样

Z_p 的 x :05AC FDD23B1B 59A24D02 363BF255 89B3F08D E45F32F4

Z_p 的 s :16DC 3A23B34B 4D0465BB 1CA6C6D2 0690493B 1DAFCDA

Z_p 的 t :20A1 A8210004 96704140 02270582 CE004557 0C440260

$X = (X_x, X_y) = [x]P2$:

$X_x = (x_0, x_1, x_2, x_3, x_4, x_5)$

x_0 :0B70B1 82A13265 72E2314C 207436C3 C1183CF1 73BEDFB5

x_1 :0B1892 3D86DCA3 A6A90A81 42ABE80F 21C9C803 87EF7294

x_2 :040F89 8B234990 4030321E CFB22E13 2D8871ED E63F1876

x_3 :0B924D 0118CE76 81611993 AF46A620 DBE38DB1 4ECFBC2B

x_4 :0398DF 11C45EB2 53D79C0E 3059E85F 1C81BAD1 FCB69ADD

x_5 :032A13 D10098C8 A06FC746 E2F19493 35426E4E 54063B1E

$X_y = (y_0, y_1, y_2, y_3, y_4, y_5)$

y_0 :05F679 664B9E16 9E15E3CE 7B2CF3AF 9A20BED7 D8C9CB0E

y_1 :06B3FB 57B4C913 C264FFA8 7D92D09D C9FA213A 78A39C51

y_2 :026A0C BAC975B6 02DE69A9 A04DE95A C49B0C5A 5624EFA1

y_3 :00FC92 2C990193 B0D3E6F3 62E6F5E7 A9CDE899 E266D240

y_4 :076E0A 64F8FB71 A2D9129F 6D2FBAF2 364FFFBC E2FFBAFE

y_5 :05D378 AB2DC3C3 8DF990E2 8B0C6827 0437D621 BF6A1506

$S = (S_x, S_y) = [s]P3$:

S_x :06CA FE1D2A1E CE257D78 3A49C0F6 C5FED69C E4180B8B

S_y :0EE0 D73E74E1 BF527463 92DF4A60 24018C27 345E9B79

$T = (T_x, T_y) = [t]P3$:

T_x:06F2 FD1BE528 55EFA7B3 303B879F EBE4CBD5 887727EE
T_y:13BB 40A5AE4B F3EBAF0F EC8A5CC4 A9D558C0 8A7A0E68

成员发布过程:

签名方群组成员凭证(W,A,y)

W=(W_x,W_y)

W_x:03C3 C8D80259 478593F6 852FD89B DE8E09FD D0B9EBD8

W_y:058B DF242DAA 341657A2 48377F99 3A65A35F AEB3CE5D

A=(A_x,A_y)

A_x:0CBC07 77BB38ED 6BCE1C08 D6C496BA C5CDF729 F3E0CCCC

A_y:0936D6 94EAF1C2 15FB3D1E 5851C206 5111E505 32C36108

y:1397 D0B12011 A8DA199C 535A908B 81E5DBF2 54BF8B9F

签名方群组成员签名密钥(f,A,y,z)

f:0C76 87681421 9C250D38 546DED0F 253525C4 E84366DB

z:0347 6076F520 2944FE03 6E40FE87 D886428D C574E6A8

签名过程:

消息:符号串“ABCDE”

群组签名(B,Z,V,W,c,f',y',z',h',g'):

B=(B_x,B_y)

B_x:07912F EF36C214 DC913C2A 3C3A9BB0 127F560E D1D80E49

B_y:07A9C7 61A1BAF2 FBEBE97E C211BC60 CDD5E0D9 DAE6225C

Z=(Z_x,Z_y)

Z_x:0166 3AD7DD71 697B2DB1 7A559F56 AD1F0548 7DDC9C12

Z_y:0F3F 47B1D570 F2DB2EE4 84DED8C3 32AA528C 261B14B7

V=(V_x,V_y)

V_x:1AB9 E279BCBD 3E5224D2 986F2D1A 4F6F98E0 C0CFC1CF

V_y:16E0 9280CFF0 869E973A 1555908C 8BA5305E 3150C947

W=(W_x,W_y)

W_x:223F AA0A39B4 A638C216 8DFC88D5 FF936051 2027F8ED

W_y:05A9 3A2223F0 D6113843 2C068B57 1B69106C CA2B0F64

c:C9AE83A0 56AD631E 53375DDF B9E894C6 AC0E97E1 0E3FB4AD 749E111E 88EFB250

f':2201 7C476D90 F12011AC 8BFC71A8 67CAC1B3 488F7B26

y':19E1 C8D50B48 166E7503 B0EAAA30 4D9C2207 42510017

z':193D FBF8D19B 78537DB2 69C2BB5A F934C3AE D00FDF0F

h':0585 252EEF16 711A02B9 F586144A B85F51CD C7214BFE

g':00FB 4EC63FE0 7CEB2B1C 8735814D 39250BD1 E1148F02

杂凑计算时注意:

将下列数据的联接输入到密码杂凑函数中:

(p,P1,P2,P3,X,S,T,Q1,R1,B,Z,W,Y,V',W',Z',Message)

数据的值除了“Message”分别用编码的 INTEGER 形式表示。

SHA-256 为使用的密码杂凑函数。

E.7 机制 7

群组公钥参数：

G_1, G_2, G_T 使用 ISO/IEC 15946-5 中规定的 MNT 曲线构建的。让 q 是一个大素数。 G_1 是椭圆曲线 $E/F(q)$ ，其中 $E: y^2 = x^3 + ax + b$ 。 p 是椭圆曲线 $E/F(q)$ 的阶。 N 是曲线 $E/F(q)$ 上点的总数。 $h = 1$ 是 $E/F(q)$ 的代数余子式。设 $k = 6$ 是嵌入 $E/F(q)$ 的次数。 G_T 是 $F(q^6)$ ，其中 $F(q^6) = F(q^3)[w]/(w^2 + nqr)$ 和 $F(q^3) = F(q)[v]/(v^3 + c_2v^2 + c_1v + c_0)$ 。 G_2 是椭圆曲线 $E/F(q^6)$ 。 nk 是椭圆曲线 $E/F(q^6)$ 上点的总数。 $E/F(q^6)$ 的余子数为 hk 。

在下列数据示例中， $P = (P \cdot x, P \cdot y)$ 在 G_1 中表示为 $P \cdot x \parallel P \cdot y$ ，其中 $P \cdot x$ 和 $P \cdot y$ 是 $F(q)$ 中的元素。 $s = (s_1, s_2, s_3)$ 在 $F(q^3)$ 中表示为 $s_1 \parallel s_2 \parallel s_3$ ，其中 s_1, s_2 和 s_3 是 $F(q)$ 中的元素。 $t = (t_1, t_2)$ 在 $F(q^6)$ 中表示为 $t_1 \parallel t_2$ ，其中 t_1 和 t_2 是 $F(q^3)$ 中的元素。 $Q = (Q \cdot x, Q \cdot y)$ 在 G_2 中表示为 $Q \cdot x \parallel Q \cdot y$ ，其中 $Q \cdot x$ 和 $Q \cdot y$ 是 $F(q^6)$ 中的元素。

$q =$

8EB4FCF5 E831AC7A B5918D6A B6226E7D 3A7CE961 3A2FD63F E2F86811

$n =$

8EB4FCF5 E831AC7A B5918D6A B621AF5A 6691BB9C 34648F22 59467B0D

$p =$

8EB4FCF5 E831AC7A B5918D6A B621AF5A 6691BB9C 34648F22 59467B0D

$a =$

11C4B1A7 E60AF161 46FBDF11 4208D560 D241D879 7FE9A229 4C653A70

$b =$

5D038BB0 181FB159 62AA4DC1 FDF13317 8A8DB587 90C8911B 8A0B8B45

$k = 6$

$nk =$

07AE945E 242E0A64 26BE3DF6 A0429809 F8EA88FF CB19B28B 9341CF5D 54ACD9EB
4C802EA8 6BF986B4 DE76A51E 27248848 4F75C578 AA9896CC 7A378F72 9BDF5A568
D99309AC C28649A7 473595B3 7C0FFFC2 92A2A888 A63E7B0A 1DDF7AC5 1E07B3AC
35C03A38 EB92A309 2168B3FE EF87716F FDCA808C 1F7C2B89 14ECA69E 610CBF9D
C5DCB366 6020E3CD 2B23C573 A5584BC2 F8B6AF7C 06A95C4F C4ED54D1 A5147A01
19045A01 621293C0

$hk =$

00000005 18B880C5 4F1AE5B0 BB7B6180 1B7583B0 158C3049 491A935E 046AEF5A
83631981 6C28A3D3 64E63620 C3FFC435 9735DF43 88233803 F3777E38 DF54040A
16D6A28B 2927825B 7D38C61F 0014723E F0095BCF 30E87052 DD4F309D CA0924D6
564B992A 0F785228 E9EA9BB9 A6EB640A FD7D0DC0

$c_0 =$

71B61292 294FE6E9 9ECBA992 568FDA9E EC5A66C2 E9C443B1 F25D21B0

$c_1 =$

7F8DEAAF 20A4F9EF 2BFBE042 4F5D9A2F 07F4FB13 2BD20C38 818F9253

$c_2 =$

4F12F1E4 E22E9625 82755E9D 53631D78 076A49DD AE4AB528 1B09D79E

$nqr =$

015AEFB8 F1373F77 95160D17 CB335465 5694A760 F35E0C5B 0F81FADC

SHA-512 作为基础的密码杂凑函数。实现特殊的密码杂凑函数使用附录 B 中的基础构件。使用优选的 Ate 对作为基本的双线性映射函数。

密钥产生过程(建立过程):

$B_1 =$

2C07EF74 006B0CF8 12A1AAB5 C15A5CCB 75CE2D3C 9B6A8126 06120F7D 05D44569
D886A9E2 53D7B60B 5698A1CC B3E8644D 9E16216E 417DB8DC 6CB09632 8447C608
A1E876B3 B884AA06 3240A6B6 3E2C146B 0A5CC29F 19175F9A 996BF730 479C6474
551C0A56 267E61E0 256BD2D6 887F87B8 2EC5ABAD EAOE5D6D EEOF2FB8 6D02EF2C
E5FACC4D 8D1F9C19 06E12A01 7952A417 17B5858B 4601098A 2DE7BB89 6AA5B90E
0091A492 A6D8CC34

$Q_1 =$

372F0C0C 88F994EC BCAA34A4 EDA8B38E 2BBCDE3F 1438208F 522A7C4F 7967C68E
5687DDCF A4BECA0F 3A00B94C 4308C2CC 071D0441 EF651D5D

$Q_2 =$

8937DDC5 2866D769 BCBC2887 D1A6C8A8 4C4CEB6D 487DBC0D C4B374C6 667C91A4
10CDB7ED B587DE87 97BD48BC 510F0B10 9DD5D764 A940139E

$Q =$

3E90AA03 F2392E8F 9DD434D4 92989798 262E6BEA 7880EF7E A8802776 171C83E2
6CAFA2B6 3C394A6D 1BF94F51 3AB8A247 8C0A11DE C60BFB81

$U =$

558E9871 45C7D9F4 D228B8A6 3BF363B4 9E6ACA1C 23841DD0 4A59820C 6EDAAF04
DE418700 4798A851 0E766EDB 1F615854 C75AC203 8300E212

$\eta =$

1F994E0E 748A8898 E09434C2 EAC7DE40 874204F7 DD65AC0E 91380872

$\xi =$

43A96A53 20694C9F EE2404A9 573CB995 AA3C0563 A31FF244 5DE21482

$\theta =$

6093B251 D6F7FCBD 82C41A4D E8C54BEA A22B4D96 565B0C99 3CFBD90B

$W =$

8D4E50F6 EBF2DDBD BAE6592D E9E01518 88D40527 B9030AD1 F6D93165 49AF9E62
2B267FAA 00ED30B0 78B89413 D87919F9 E5F6B1E6 D59FA77C

$D =$

7CE8A928 CF0C2D34 489F5B9A DBA45CD0 9D9D95D4 9F75F810 64CB0B89 58ED9268
80B826F2 DC6CDA5D 4B33C74C 2FBD3BAC D3A2317E 03D37E1C

$B_\theta =$

63ADFD0 B2197470 E1489589 A1C807EC 24DF5DD5 6421CF6A E1577C90 5832857A
E31F1087 DBFFAD8E 1F1C63BD 94F4B38C 41170703 3CB366D1 602F1583 4AA64751
284D4B2F DBE74425 E19256D5 7D8D8526 74F39721 77A7174D 073F9DBF DE2962EF
0F766E39 4E07B3C0 40D2418E 0EA862CA 2DBA4CD7 ABC13E80 6184AC33 D03F4536
10C8A366 2BA14B1E 0ED2F013 4393C904 CEC7E268 BF3EADCD 72CD12C8 EA78EAD5
0CB4315A 12B9627F

$V =$

56235E87 DA187995 E7AEDED7 D0D785C9 2867C74B 16170E06 7888AD26 57E45E56
 7552BEB8 BE6361D3 D27DC2DA C065749E 67B9B978 95750EBF 89F4947B 144FBA5E
 AC5FE697 EC6AC03A AB2DCEB1 47AC1E9C CC60F1B3 12E666D8 C5551116 4163F939
 9BDA8885 2BAC36C3 B6E84D59 81058C7D 4C1BB251 54E7E174 8FBFB CAB 98EED5D2
 97480D14 E0FAD647 A8F0166D 5D3E43F9 92D0E617 3492B575 9C6112FD 93D0B104
 FBF72C46 C06AE094

$L_1 =$

335F307D FEFBF835 011934DF 3E00C051 9AE6E0C6 59ABCEE1 A1F207F6 2D589F4C
 65F78D46 D100F981 0E0210C4 308E519B 57E15312 212D6281 0411A00D D0D9392B
 B7F7D6E7 1AE9E832 2577E419 E117F7A6 66AA94D0 291B18AB 6E8ECEAA 077067F1
 999AD861 B95D52B0 C47E4AD2 2BFA2C07 68853C34 4D6D9CC8 C73FCF9B DE4705D0
 AAEA479A 3E13E8CC E29F5CE9 12368E3D 6B63A4C7 3D8232A3 1328C30E 6FB6A875
 72B2520C 5AA6B77F

$L_2 =$

05A28AD6 1DA904C5 2B12F6CE 63AEA29B 03B759EA C22BD8FB 1D330B6B 8A221E0C
 A57A5056 EA1A07A9 A315A01F 35302D81 96087B4F 06963D23 011A393C C5FB8316
 8F48AF70 8ABD7053 864D58CA AA3F517E CE301465 20448BCF 6309748B A8197CFE
 B11CF4BE 24E166BC 8C0627F0 4CD95650 82749A81 F04E937B F9484EAF 88DAB0A2
 31605ED3 81D16F7B 37AF0DA6 847EC67F 8B0B03CD 0113992F 3A70BFC3 527243AC
 1965F8DA 5CFC229E

$L_3 =$

88EB727F 63BD1104 1A86969E 363C8419 5D971C64 5674002F A878B944 76141DB7
 656E65F8 5203A7C3 D7AA4322 C2B36602 304F98AB 070DB487 5C282208 27B5B265
 1B054BCF AC6475D3 BC6F6BD2 0F6516DA C57EA9CB 415C17B8 080623B3 FEF1A940
 1A5842EF 536591EA E9449D66 14378ED5 68B11685 6829A931 557D0D6A D13A864C
 5A4CCDFE 2CC700F0 8AE8369A 7253A0FB 93B860D0 F9D5B8DC 73F630A2 3BDF6B28
 2CB1DC41 FAF7BB34

$L_4 =$

65DE3723 D62A06A5 AB78AD5D D23E7641 CEF15F20 46F4E6B7 0B3050DF 80E8AC1E
 4FE8E186 8B9A71A8 9F0C4FC9 A33C99BF E139C993 916639FB 7348CFEA 51BC9020
 B412580C 7E0FF3DC B847F390 D6EBFCA2 9E95381B 0357B9C4 BD7610DC 9DF9B87B
 AA7E4026 1EC6DA8D A3D94C3F CAB8D353 2A30652A 894BBC96 53F09F07 6C8966EF
 44283CE8 A1CBA7EE D9B377E1 778B1596 1A984F66 1739BEED 808C7869 DA423785
 F045587F D369CCCO

密钥产生过程(群组成员发布过程):

$ID = 0001$

$sk =$

32711F23 7AF94DEB 4F5F7864 95FC75BE 21FE0ED8 23809AA7 C04BAF07

$Z =$

036ACC66 9A29C7A9 C990FDCE BE3941DC 2167FCA0 72FBF996 5097ACC8 7CECC4BF
 AD3CAB66 36D365A9 4A79D1DC 6016B7E9 66CBA09E 24B3A963

$r_{ID} =$

3BA65FBB 46903233 3DF8B310 DB666F62 CF3235F6 7C5479B8 2404A7BD

 $W_{ID} =$

3FCB4576 C2668452 FCD48D45 ECD91D3A 9DA787B6 022905D1 F83A6AD4 0F6D72D4
916A0641 DC730EF0 AF5ED149 4E528092 9ACD574B DCD4904C

 $c_{ID} =$

15FE5B75 0A8642AB 1DED7BC9 04F12F54 A6314A8F E5A3BBAE C4774934 7A8D59E7

 $s_{ID} =$

810E2FD2 D473568F 5E8E9363 9B3C2948 FB719259 A3578324 DF0636D9

 $T_{ID} =$

036ACC66 9A29C7A9 C990FDCE BE3941DC 2167FCA0 72FBF996 5097ACC8 7CECC4BF
AD3CAB66 36D365A9 4A79D1DC 6016B7E9 66CBA09E 24B3A963 810E2FD2 D473568F
5E8E9363 9B3C2948 FB719259 A3578324 DF0636D9 15FE5B75 0A8642AB 1DED7BC9
04F12F54 A6314A8F E5A3BBAE C4774934 7A8D59E7

 $x =$

4A8B2691 3B8C143F F97C6B51 D628A10D 85C20645 6689E7DF FCC6A0ED

 $y =$

825E3540 5EDC0AB3 AFF27AE5 7C70C12B 071D158C A927212F CA89C5A4

 $A =$

661FB3A5 6E8E2169 41DA69C3 3B4C68A5 4DE93A50 D6A70D38 2FC4FBEF 1E69E40A
0C9FD75F EE4F18EF 54B684A3 31183C86 07F8810C 9742F4A8

签名过程:

消息 $m =$

“abcdefg”

 $\lambda = 00$
 $\alpha =$

417A526A A65ABA2B 4D157B5C 23A3986D 80538DB2 F41F1F70 30106E96

 $D_1 =$

6E0B4988 949D569E A38BABBB0 DB22ACF0 66FC528F DD8BED53 E28903BF 5C2E1378
3599126E 408242E8 5256B059 65ED8D48 CFF08E1F 7D0485CF

 $D_2 =$

40D1AEF3 7959B516 5CCB59DF E3CF3A26 45789F64 49447368 A772950F 4431C2F2
C5D0BA0D D21C2149 7414E8A4 6C1F30FE 7FD862A5 F5A064CB

 $D_3 =$

60BBD7E2 A2965EFD 30DAAE91 D6B34F77 2F9D36A7 8F70B2F8 F22D4389 25A851E6
D5AFAAD8 FF91F2BE 877517AF B86E7723 615B1F51 B9E64903

 $\gamma =$

11072441 7E2BC0BF FB6B4B4A 790439AC 43508D3F 3FADB60C A406924E

 $r_a =$

2F7AA3BB 8D9DBB29 B175E949 F5506C76 975FA3D3 A6D3351E 5A1E30CB

 $r_x =$

88C396EA 142B6DAA 6407AD99 8D336B06 14978BA0 40E69185 A7DB2E27

$r_y =$

68CC8685 618999DF E6CC367E EAF0D3A8 E8BF08D6 B7A6E1CD 1C2375C7

 $r_y =$

1A116AE6 1C5EB9BC B29FE989 6A285E4C AC532506 6D9BCE57 1D2BFF21

 $R_1 =$

14F8817D 08690636 403B6467 A05B56A6 0894BF79 00E40D6D FC47DD58 2371916A

343FE67B 5DBA2CAC FAE33268 BD9749DE F8A85E68 5BB971C5

 $R_2 =$

80824C69 17AEAF74 4EBA35B3 6E6F2C66 2E9449F5 17BA47EB C3194B6A 0FC56B8A

4211492A ED4E117F 5B9A3C7B 641E5DF2 4A7924B8 BE54ABC5 3D84996C B1770F21

ED9DBFDF 3AC95C56 838A2582 BD2BF992 AD1B22F1 7BD00EA1 45452815 87EC3A9B

E7E9C706 4E00D4A1 9EFF0E5A 81DECDE7 8353F70D A47912D7 1799A5D9 2C943B4F

4626DDAA 69A71910 CC199310 808C26F6 AA142802 92D86F06 2AA9DCFA 40076F4F

DA7398D2 9ECAE30C

 $R_3 =$

42D6C598 D5AA7423 7C6016D5 6C036E3D CB1C4B55 0892DA3D 65741798 00AE96B8

3CA07FA6 99222D51 689EF22F 26DA1B79 7EC10C1C 0CFEA475

 $c =$

D600071F 11026E11 7841646F 16239A97 807C83C5 2223166E 7EE79095 0A18C2AC

 $s_a =$

4C55B7A5 17F9B840 077CB66C 65F7D71F 28FCCA53 F9F2790A 088E8F3F

 $s_x =$

4D04B081 48E2D699 17F19D1D 4A70E6B3 D6D9AD7E 6054B82A 9CFA747B

 $s_y =$

5C3B319D 13F129C8 644BEF87 9B56CED0 77E84944 DCA7BA49 6507304C

 $s_y =$

47D2B829 1128380F 0EAC2E98 95FEAD0E 51D2A541 E81E6D9A 77C47EEA

验证过程: $R_1 =$

14F8817D 08690636 403B6467 A05B56A6 0894BF79 00E40D6D FC47DD58 2371916A

343FE67B 5DBA2CAC FAE33268 BD9749DE F8A85E68 5BB971C5

 $R_2 =$

80824C69 17AEAF74 4EBA35B3 6E6F2C66 2E9449F5 17BA47EB C3194B6A 0FC56B8A

4211492A ED4E117F 5B9A3C7B 641E5DF2 4A7924B8 BE54ABC5 3D84996C B1770F21

ED9DBFDF 3AC95C56 838A2582 BD2BF992 AD1B22F1 7BD00EA1 45452815 87EC3A9B

E7E9C706 4E00D4A1 9EFF0E5A 81DECDE7 8353F70D A47912D7 1799A5D9 2C943B4F

4626DDAA 69A71910 CC199310 808C26F6 AA142802 92D86F06 2AA9DCFA 40076F4F

DA7398D2 9ECAE30C

 $R_3 =$

42D6C598 D5AA7423 7C6016D5 6C036E3D CB1C4B55 0892DA3D 65741798 00AE96B8

3CA07FA6 99222D51 689EF22F 26DA1B79 7EC10C1C 0CFEA475

 $c =$

D600071F 11026E11 7841646F 16239A97 807C83C5 2223166E 7EE79095 0A18C2AC

E.8 机制 8

群组公钥参数:

本机制采用 GB/T 32905 给出的密码杂凑函数,其输入是长度小于 2^{64} 的消息比特串,输出是长度为 256 比特的杂凑值。

本机制选用 BN(Barreto-Naehrig)型椭圆曲线。所使用的双线性映射函数为最优 Ate 对。

本机制中,所有用 16 进制表示的数,左边为高位,右边为低位。

本机制中,消息采用 ASCII 编码。

安全参数: $l=256$ 。

椭圆曲线方程: $y^2=x^3+b$ 。

曲线参数:

阶 $p=$

25236482 40000001 BA344D80 00000007 FF9F8000 00000010 A1000000 0000000D

素数 $q=$

25236482 40000001 BA344D80 00000008 61210000 00000013 A7000000 00000013

系数 $b=$

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000002

余因子 $cof=1$

嵌入次数 $deg=12$

扭曲线的参数 $\phi=$

25236482 40000001 BA344D80 00000008 61210000 00000013 A7000000 00000012

$\xi=$

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001

$F_{q^{12}}$ 的 1-2-6-12 塔式扩张:

$F_{q^2}=F_q[n]/(n^2-\phi)$;

$F_{q^6}=F_{q^2}[g]/(g^3-\xi)$;

$F_{q^{12}}=F_{q^6}[d]/(d^2-g)$ 。

群 G_1 的生成元 $P_1=$

24A01831 6C547AED 5A6B52D7 FF66066F A9E9893A 4A426735 650F0DFE 2467E706

13D06919 CC217953 C2A13DBC 21CE81EB E8921FAC 33FAB18D D65B88CA 3BBA1038

群 G_2 的生成元 $P_2=$

1A03FD47 A3FA9AD7 FCBFBCB7 C28E4AF9 DD1EED86 7A68EA63 9FB8203C E31CD2EF

228FCD53 7F5E0523 60E708CF F73E3659 FF4BA332 1777522F B65718C7 1ABEA290

1CE8E13D 948F2B39 18DA1884 F37D48DE 185E460B 2D7E739F 83088A2C D169E239

07671F18 C1AE55B3 27F1C997 49E00F49 2086F4FD D4728129 9BCD70AF FAE9FE6A

群公钥 $X=$

05E97FB2 3C3CCE50 7CE47F8E E5706958 02AA7C9D 608627AC 4E2212E3 5327BC7C

1DC7B80C 20194593 A6ABFF76 F4849BA4 A455F8D7 7571D54A 4C8E21B2 3BC1B3EE

04FC1AF8 AE3F564E 155295E6 761E2748 032E043E 44C7D824 7EF884D7 485FED95

148CDDF5 AA5F9EA7 1A3A8D43 94FAA6CF 5B0E4691 C0CF9CEA B7834794 EF2D0F7E

$Y =$

227061FD	B04F26EA	D5FFDA9C	CD4F3A62	1517E222	E94EC018	EBAE4455	926DF1E4
2337EB5F	00B2E0B9	E30BAEA7	E3488A89	4C55B725	08871DE6	9135406A	C43B1A9A
0830F70D	04C51D0E	27661F7F	C01E4FA8	422E80D8	E2452DEB	7A800EB1	D1F619A0
0AE0B903	99864C01	048643A9	A4BA7999	BDD85485	C264EAB7	2BCFAE51	85D3FD06

群成员发布密钥 $x =$

168CF8A4	FC2C4D03	7E970748	4C88CE24	96A028F9	98D8704F	2A1BBAE1	DC8B3218
----------	----------	----------	----------	----------	----------	----------	----------

$y =$

24DAF9D4	5C6C7913	D9CA0405	9A8484F5	AE075CAE	E54A178C	82555753	571A6653
----------	----------	----------	----------	----------	----------	----------	----------

群打开方公钥 $Q =$

15DF076A	55191A72	1F23718C	ECFF6A27	EBE02D62	32EDF546	939B8775	D9DC5B45
04310AA3	FFEA77CE	9F12EFAF	51D96473	BC60195C	DA671FF9	367D3E01	448EEC34

$V =$

17632070	35F61600	807A5635	09F51490	D5045A9B	29346E02	A1EDFE40	820D35C3
0C285F5C	F7216427	E8214477	EDCF5AB2	88787508	8D04A496	93E90866	C159EAAC

$W =$

1887F583	726539D8	8716AF94	65973578	BFCB3039	44AC9850	AEB40572	6C5DBBE2
1065270F	EDE35C7F	A07CE2D6	DF92AF7A	D2D6FB39	39E32F8A	9AFACED6	3E5513D5

群打开方密钥 $\alpha =$

21916727	550DEBBE	3245FB95	7A57E686	068B4592	7B841425	160933D7	45549779
----------	----------	----------	----------	----------	----------	----------	----------

$\beta =$

13EA4ADB	5ABBBBCA	57D32219	8CD7592C	E57674DE	757D415E	0B650A9A	6A9FCD77
----------	----------	----------	----------	----------	----------	----------	----------

群成员加入过程：

$n_1 =$

D93A8F9B	468B6335	976C7738	29D7FB7D	7E078575	CDEC738D	984BD696	0F2B9E63
----------	----------	----------	----------	----------	----------	----------	----------

$f =$

1E41EC0C	08DDB93E	103E4406	A7CFD859	3E43D905	1AB56E7C	1DD238BC	5BD9E5E7
----------	----------	----------	----------	----------	----------	----------	----------

$F =$

23F98E49	89B3D4A4	F4EFDAE6	19FADD65	93692C1B	F4217FC8	6526B4F7	B8AE4778
0F0DE60B	B7502B7A	3B92941C	ED211EE9	39F85EEE	2553BE77	C167E577	0CC25770

$u =$

1632E56A	E36E6F1C	B02BCD3C	267FC065	F9A3494F	5D871624	138DA020	4E226F46
----------	----------	----------	----------	----------	----------	----------	----------

$U =$

1E88594F	D99382FF	75089908	5F395746	166095CC	51BDB683	3A9B03D6	094AD613
1FFD8191	969A3A46	4C7282A4	D02AB45F	F816BE3E	BB9DCD49	F4D65144	EA92759B

$e =$

0DE350BB	5A43C8E7	7483F8B8	7F4A7884	C9B38F0E	BF325C13	8F9AD5ED	B5CB5715
----------	----------	----------	----------	----------	----------	----------	----------

$v =$

07484588	F3D74BE5	2F584440	DE83CFC2	E074A4DB	10F01286	2935D9C3	BF162D1B
----------	----------	----------	----------	----------	----------	----------	----------

$w =$

02063652	DF325E00	FA87A9A6	006C5639	BB86423D	F1F9AD9C	E980CE0A	A9A6E601
----------	----------	----------	----------	----------	----------	----------	----------

$e' =$

0DE350BB	5A43C8E7	7483F8B8	7F4A7884	C9B38F0E	BF325C13	8F9AD5ED	B5CB5715
----------	----------	----------	----------	----------	----------	----------	----------

$t =$
 094E7BDB D309A9E6 29DFEDE6 DEF025FC 9BFAE719 02E9C023 12B6A7CE 68BD131C
 $x_U' =$
 1E88594F D99382FF 75089908 5F395746 166095CC 51BDB683 3A9B03D6 094AD613
 $y_U' =$
 1FFD8191 969A3A46 4C7282A4 D02AB45F F816BE3E BB9DCD49 F4D65144 EA92759B
 $v' =$
 07484588 F3D74BE5 2F584440 DE83CFC2 E074A4DB 10F01286 2935D9C3 BF162D1B
 $r =$
 14022591 54192245 9ADBEE2D 24B30EA0 F4CFD600 FD70B4DE FE7D0276 997085CC
 $A =$
 1BBEF7CC 6DEB0024 0C83E22C 62AA5390 7C8E34E5 0054A22B 14C73D98 6AFE455A
 245B8674 0D00E4C8 5E07C9E0 ADFF2868 3695E70D 8B6AA2AD 634CB7C5 C793DBB2
 $B =$
 0280F553 5B2638D1 07117E96 80DDB2FC 9E84CA2C A2CC33CC 1F27D7BC 519425AC
 050315A4 DD908889 8076E7DF 6E494CCD 0288378B 273496D1 89CC35FF FCB570A1
 $C =$
 0551C9DD 17624001 1F7718CF 50DEAFB7 86F5B72D 1A686468 8CD8EA3D 2632944F
 06FEFEDC 51B8035C 66135E74 ACDD00A2 CFE35994 76747F7D 66E762CB 7B2CAAAD
 $D =$
 2344BB54 40409977 A5A25F64 F2A2803F 34FDD702 4A665987 3E62C37F F5881F07
 0ADF2668 46C7F1B3 13E080A5 597B1F07 A246BAED 3A711CD7 CCD81B0E D4503639
 $\alpha' =$
 13292BFE 68EB2E1E EEB6E35A CB17DC86 EC70BAC1 ABBE3700 40101CAC E9FE9913
 $h =$
 0A2167DC 01FCC5D8 8FF4081F EA9B9E75 FADAED29 A4F4C58C 14152E31 3298AB4A
 $R_1 =$
 0305DA6A FCA0AE41 C9BDE9C6 FA68FF78 041F8D3C B2D39B27 077A4AB6 31B97E99
 1CFFD24F 4FBF57B7 F4A5C2B1 1FD004B3 35D47835 67928B3B D3847E3B 77F7C0B1
 $R_2 =$
 19EFE354 DE76E65B 485B874C 60EC01FA 78F66D75 06A04918 652A6DF6 AD5F8589
 0B65ECE1 B2E77040 CED7A739 04B16123 35DF809F 1F174E08 52BC4BCA C80C097C
 $e^* =$
 22EC1FA5 A16EA7D3 D1051A78 C224EB5D 2A3F3876 02582990 8FCD515B 00F8C605
 $c =$
 1ABE78E3 3C863C6F 28EA3E0C 1D79ECC7 A7B5B327 BBCC0DBF 5B720A07 E011CA1A
 $s =$
 111CB292 1EB31FEB 9D1BF2D3 0A0126E8 7962076D D2C81C8C C138D834 1FC2EF70
 $t^+ =$
 06B7C6F3 1B395C59 0BD1E35F 277B13A8 21783A95 8E942A3B 7BAAE23B FFD4B97D
 $R_1^+ =$
 0305DA6A FCA0AE41 C9BDE9C6 FA68FF78 041F8D3C B2D39B27 077A4AB6 31B97E99
 1CFFD24F 4FBF57B7 F4A5C2B1 1FD004B3 35D47835 67928B3B D3847E3B 77F7C0B1

$R_2^+ =$

19EFE354 DE76E65B 485B874C 60EC01FA 78F66D75 06A04918 652A6DF6 AD5F8589
0B65ECE1 B2E77040 CED7A739 04B16123 35DF809F 1F174E08 52BC4BCA C80C097C

 $e^+ =$

22EC1FA5 A16EA7D3 D1051A78 C224EB5D 2A3F3876 02582990 8FCD515B 00F8C605

 $c^+ =$

1ABE78E3 3C863C6F 28EA3E0C 1D79ECC7 A7B5B327 BBCC0DBF 5B720A07 E011CA1A

签名过程：

 $n_v =$

EABDE96B 45AD2162 41E5FE3E 521915AF 0D648935 289C593E 85C112D9 90AFA487

连接基:anonymous signature linking base。用 ASCII 编码记 bsn ：

 $bsn =$

616E6F6E 796D6F75 73207369 676E6174 75726520 6C696E6B 696E6720 62617365

待签名消息:anonymous signature scheme tests。

用 ASCII 编码记 m ：

 $m =$

616E6F6E 796D6F75 73207369 676E6174 75726520 73636865 6D652074 65737473

 $f =$

1E41EC0C 08DDB93E 103E4406 A7CFD859 3E43D905 1AB56E7C 1DD238BC 5BD9E5E7

 $F =$

23F98E49 89B3D4A4 F4EFDAE6 19FADD65 93692C1B F4217FC8 6526B4F7 B8AE4778
0F0DE60B B7502B7A 3B92941C ED211EE9 39F85EEE 2553BE77 C167E577 0CC25770

 $Q =$

15DF076A 55191A72 1F23718C ECF6A27 EBE02D62 32EDF546 939B8775 D9DC5B45
04310AA3 FFEA77CE 9F12EFAF 51D96473 BC60195C DA671FF9 367D3E01 448EEC34

 $V =$

17632070 35F61600 807A5635 09F51490 D5045A9B 29346E02 A1EDFE40 820D35C3
0C285F5C F7216427 E8214477 EDCF5AB2 88787508 8D04A496 93E90866 C159EAAC

 $W =$

1887F583 726539D8 8716AF94 65973578 BFCB3039 44AC9850 AEB40572 6C5DBBE2
1065270F EDE35C7F A07CE2D6 DF92AF7A D2D6FB39 39E32F8A 9AFACED6 3E5513D5

 $r' =$

14F822A8 0841F7F9 B44E9030 86D6C2A9 D6ED4276 AEFE40B2 E38477B0 63DEC24D

 $A' =$

0F1BEFDB 1938D44C 502EB2EF EED0139C BE220FB9 02089836 82C85542 62485D92
1E36328B 3AE89820 03BD54A4 61A6A3A7 C666816E 0FE887BE 0950D18A 3A1E1B2B

 $B' =$

179FBDEA 61E2A861 3954B057 AE0B349D 00FD3524 425E8183 78CC4756 EF6832B3
10F400EC DD8A464A 0DD2190E BF9CD722 AF7A345A B315FFE6 45042D63 85100C8B

 $C' =$

11F8E89D 5515D08F 142C88EB 610B6FD2 EC794638 CBD9B27C 6DCE3027 076F79FB
18BB9C7F 6B9F382B DFD3D3EF 040FC915 62086EC0 CC139B12 ACD5CF28 D13AF7DC

$D' =$

201EDE32 B7DB694F 1A7605A7 671E9B7B 61E66F76 0AB55C73 E68152D5 C82DF90D
05DA544A 94050832 7EB064B3 01E7E6C2 B9C33F49 2FF36942 192DBF6E C2D0CE44

$J =$

0FA5C1F6 BB33C4B4 0B65EBB8 D3FE6587 C6BBEB6F A009D4CA 17B16E9F E9DD162C
097EEB5C 55D1F936 42FE9AAE 6A66D98E B4B45F0D A1212CFB 2703FF18 DF048381

$K =$

11C04E89 285112BA 8E95EDD3 40E1D2A8 035A4BB0 41F16CCE A366A4B7 C41BD40E
224FA178 3348E13E B0376615 C26750F6 DB94941D C67C2959 0AB166BE 26FB00AC

$r_a =$

2347BD70 320CE970 E741ED9B 3CDB2423 DD1029B6 E963378D 2A22283A F91D77FA

$r_\beta =$

10CAEBD0 F1A542A4 81A13624 AA2F7899 EB804A66 A873F3B2 D228AE21 95EC0496

$C_1 =$

218D5D2D FD51EF52 A590AE6F B8AFCA66 1ED6DE8A DCA0982F 60874B49 6B02B504
1E628EC9 4539A0F2 7A3FD908 FCD6BB60 6F31D485 663E4B0F 47339A09 94AA8275

$C_2 =$

0CDD1A6E D82CF4FA C70D6727 85AB2788 21888FF1 76576831 F0AE1CFC B0DC6E59
17C2B3ED 7C1DAC7A C0B9CF6F 1E55759B BD5274C3 B9F237F7 C311D82C FB90AA9B

$C_3 =$

0847C7F9 3F027E9C 31CD9D74 C40BA5B6 E3B46584 056F2797 2B84DD90 50F8F639
22015C37 C39E4CAA E1172600 50A3F3FB 7D7FA76A 99B0D846 209415F7 2D4FE487

$a =$

24170C28 98AB9A73 32BC873E 3715E434 4B3454A7 32433FEB 71AB2584 9067C7F5

$R =$

13EA4E52 E86D3E30 148177F0 C2A91877 6050AF52 8E7110C4 0D8C761A C1D340DC
134D57BC E1D74271 FF171F23 1A4C6A97 633A92E7 AD0F2303 95C698A2 8C2AB156

$e_1 =$

0EB0FC5B D7BEB22D 60A97109 17922118 F8827D8A 2AF4C858 B839F5A7 AAD88ED6

$c_1 =$

229B4AAE C02BF05D 752AE8F9 DA3B3990 58D32CDC B965D91C C5C66BC2 6CABCFB2

$s_1 =$

1B823644 9832EDCD 056CB084 0D409AEB 8DD215C7 C1D87FB4 3F41C5F0 21B10116

验证过程:

$e_2 =$

0EB0FC5B D7BEB22D 60A97109 17922118 F8827D8A 2AF4C858 B839F5A7 AAD88ED6

$t_1 =$

18FA1C71 185EDE28 C0634BFD E77BD473 E705C2A4 7B3E58C0 640831B2 8E5CD0BB

$c_2 =$

229B4AAE C02BF05D 752AE8F9 DA3B3990 58D32CDC B965D91C C5C66BC2 6CABCFB2

打开过程:

$\alpha =$

21916727 550DEBBE 3245FB95 7A57E686 068B4592 7B841425 160933D7 45549779

$\beta =$

13EA4ADB 5ABBBBCA 57D32219 8CD7592C E57674DE 757D415E 0B650A9A 6A9FCD77

$C_1 =$

218D5D2D FD51EF52 A590AE6F B8AFCA66 1ED6DE8A DCA0982F 60874B49 6B02B504

1E628EC9 4539A0F2 7A3FD908 FCD6BB60 6F31D485 663E4B0F 47339A09 94AA8275

$C_2 =$

0CDD1A6E D82CF4FA C70D6727 85AB2788 21888FF1 76576831 F0AE1CFC B0DC6E59

17C2B3ED 7C1DAC7A C0B9CF6F 1E55759B BD5274C3 B9F237F7 C311D82C FB90AA9B

$C_3 =$

0847C7F9 3F027E9C 31CD9D74 C40BA5B6 E3B46584 056F2797 2B84DD90 50F8F639

22015C37 C39E4CAA E1172600 50A3F3FB 7D7FA76A 99B0D846 209415F7 2D4FE487

$F' =$

23F98E49 89B3D4A4 F4EFDAE6 19FADD65 93692C1B F4217FC8 6526B4F7 B8AE4778

0F0DE60B B7502B7A 3B92941C ED211EE9 39F85EEE 2553BE77 C167E577 0CC25770

撤销过程:

$x' =$

13300C0C 6FD565C4 6EDE5DA3 5F18057D F2C25D33 E336A975 862A7D5C C00F4C7C

$X' =$

1EDA75F5 330DC6DA E0CE1BDC D6655148 BF6454EF 1837B22A 83AF0A7E 009092B0

0C65B0C1 5D6925F3 4CEEF803 9825D1A3 BE3EEC63 66B62BC2 80A9E2BA 34983584

1CD4A0AB 0D534F26 C07FFCF2 B16874A7 6D6A4B44 DD411EAA 33525BF0 BAEC2032

15B61DE8 60873AAC FD903B0E FB4C76AF 3A3E3589 0F30450D 01F35488 A39DF936

$\theta =$

0B6C2CEA 2C398A3F 0E8268C7 D4AA1FA5 77BEAE1F 2232F822 15AC0800 DE711699

$C^- =$

03AB0CFE 989C0577 53AFEA0F 3E29367B 22436D91 E520C8BA 2CE13BBA CF4F63AE

1CC22AF5 1AE504FF BEA87173 44E5947D 81C7F538 81C390DC 76C698BB 447B4EA9

附 录 F
(资料性附录)
机制 5 的正确性证明

F.1 证明群组成员发布协议步骤 b) 的生成正确

F.1.1 概述

机制 5 中群组成员发布协议步骤 a) 和步骤 b) 描述如下：

- a) 群组成员从 $(0, 2^{\Gamma})$ 内随机选取 x' ，其中 $\Gamma = K_n + K + K_s$ ；
- b) 群组成员计算 $C = a_1^{x'} \bmod n$ 并将它和一个正确生成的证明一起发送。

为了证明其生成过程的正确，用户/成员应证明下列两个条件：

- a) 用户/成员知道 x' 使得 $C = a_1^{x'} \bmod n$ ；
- b) 该 x' 在 $(0, 2^{\lambda})$ 的内部。

证明第一个要点使用如下被称为 Proof-of-Dlog 知识的子协议。

为了证明第二个要点，应证明 x' 不是负的并且 $2^{\lambda} - x'$ 也不是负的。

如下被称为 Proof-of-Non-Negative 的子协议用于证明已知 $C = a_1^{x'} \bmod n$ 、 a_1 和 n ， x' 是非负的。

已知 $a_1^b / C \bmod n$ ，其中 $b = 2^{\lambda_i'}$ ，比较容易去证明 $2^{\lambda} - x'$ 是非负的。

群组成员发布协议的步骤 c) 由发布方执行，验证以上的证明。

F.1.2 Proof-of-Dlog 知识子协议

在给定基 a_1 和 $A'' = a_1^x$ 的情况下，该协议输出比特长度为 Γ 的 x 的知识证明。算法(证明)如下：

- a) 随机选取非负整数，比特长度为 Γ 的正整数 $u_{(0,1)}, u_{(0,2)}, \dots, u_{(0,m)}$ ；
- b) 对于每一个 j ，计算 $u_{(1,j)} = u_{(0,j)} + x$ ；
- c) 随机选取非负整数 $r_{(0,1)}, r_{(0,2)}, \dots, r_{(0,m)}$ 和 $r_{(1,1)}, r_{(1,2)}, \dots, r_{(1,m)}$ 。计算 $t_{(i,j)} = H(u_{(i,j)}, r_{(i,j)})$ ；
- d) 计算 $(b_1, b_2, \dots, b_m) = H(a_1, A'', \{t_{(i,j)}\})$ ，其中每一个 b_j 的比特长度为 Γ ；
- e) 计算 $v = \sum u_{(0,j)}, b_j$ ；
- f) 计算 $a = a_1^v$ ；
- g) 计算 $(c_1, c_2, \dots, c_m) = H(a_1, A'', \{t_{(i,j)}\}, a)$ ，其中每一个 c_j 是 1 比特；
- h) 设 $U = (u_{(1,1)}, u_{(1,2)}, \dots, u_{(1,m)})$ ；
- i) 设 $R = (r_{(1,1)}, r_{(1,2)}, \dots, r_{(1,m)})$ ；
- j) 对每一个 j ，设 $D_j = t_{(1-c_j,j)}$ ；
- k) 输出 $(\{D_j\}, (c_1, c_2, \dots, c_m), U, R)$ 。

F.1.3 验证 Proof-of-Dlog 知识子协议

在给定基 a_1 和 $A'' = a_1^x$ 的情况下，该协议验证比特长度为 Γ 的 x 的知识证明 $(\{D_j\}, (c_1, c_2, \dots, c_m), U, R)$ 。算法(验证)如下：

- a) 设 $t_{(c_j,j)} = H(U_j, R_j)$ ，其中 U_j 和 R_j 分别是 U 和 R 的第 j 个元素；
- b) 对每一个 j ，设 $t_{(1-c_j,j)} = D_j$ ；
- c) 计算 $(b_1, b_2, \dots, b_m) = H(a_1, A'', \{t_{(i,j)}\})$ ，其中每一个 b_j 的比特长度为 Γ ；
- d) 计算 $v_1 = \sum U_j, b_j$ ；

- e) 计算 $v_0 = \sum U_{i,j}, c_{i,j}$;
- f) 计算 $a = a_1^{v_1} A''^{-v_0}$;
- g) 验证下列等式 $(c_1, \dots, c_{i,2}, \dots, c_{i,m}) = H(a_1, A'', \{t_{(i,j)}\}, a)$ 是否成立。如果成立, 输出接受。否则输出拒绝。

F.1.4 Proof-of-Non-Negative 子协议

F.1.4.1 总则

给定基数 a_1 和 $A'' = a_1^x$, 该协议输出证明 x 的比特长度 Γ 是非负的。

同一个基数 g : 同时用于子协议 Proof-of-SqRoot 知识和 Proof-of-Loose-Range。

算法(证明):

- a) 计算 x_1 为小于或等于 x 的二次方根的最大整数, 计算 $x_2 = x - x_1^2$;
- b) 计算 $x_3 = x_1^2$;
- c) 选取随机正整数 r_1 并计算 $C_1 = a_1^{x_3} g^{r_1}$;
- d) 通过 C_1 使用 r_1 和 g 来进行 $a_1^{x_3}$ 的二次方根的证明, 使用 GenSqrtPf 子协议并获取 proof-sqrtpf;
- e) 计算 $C_2 = A''/C_1$;
- f) 通过 C_2 使用 r_1 和 g 进行 proof-loose 的证明即 $x_2 = x - x_1^2$ 是足够小, 其中 $C_2 = a_1^{x_2} g^{-r_1}$ 。使用子子协议 GenLoosePf 并获取 proof-loose;
- g) 输出 C_1 和分别输出 proof-sqrtpf 和 proof-loose。

F.1.4.2 子子协议 GenSqrtPf

通过使用 r_1 和 g 的 $C_1 = a_1^{x_3} g^{r_1}$, 该协议输出 $a_1^{x_3}$ 二次方根的知识证明, $a_1^{x_3}$ 的二次方根为 x_1 。

算法(证明):

- a) 随机选取非负整数 s_0 并计算 $U_1 = a_1^{x_1} g^{s_0}$;
- b) 设 $u_0 = r - s x_1$;
- c) 随机选取非负整数 y 和 u 并计算 $C'_2 = U_1^y g^u$;
- d) 随机选取非负整数 s 并计算 $U_2 = a_1^y g^s$;
- e) 计算 $c = H(a_1, g, C_1, U_1, C'_2, U_2)$ 和 $y_1 = c x_1 + y, u_1 = c u_0 + u, s_1 = c s_0 + s$;
- f) 输出 (U_1, c, y_1, u_1, s_1) ;

F.1.4.3 子子协议 GenLoosePf

通过使用 r_1 和 g 的 $C_2 = a_1^{x_2} g^{-r_1}$, 该协议输出证明 x_2 足够小。

算法(证明):

- a) 随机选取非负整数 y 和 u 并计算 $C' = a_1^y g^u$;
- b) 计算 $c = H(a_1, g, C_2, C')$ 和 $y_1 = c x_1 + y, u_1 = c r_1 + u$;
- c) 输出 (c, y_1, u_1) ;

F.1.5 验证 Proof-of-Non-Negative 的子协议

F.1.5.1 总则

已知基 a_1 和 $A'' = a_1^x$, 证明 x 的比特长度 Γ 为非负的。

该证明由 C_1 、proof-sqrtpf 和 proof-loose 组成。

同一个基 g 用在子协议 Proof-of-SqRoot 知识和 Proof-of-Loose-Range。

算法(验证):

- a) 通过使用子协议 VeriSqrtPf 的 C_1 可验证 $a_1^{x_3}$ 的二次方根的知识证明；
- b) 计算 $C_2 = A''/C_1$ ；
- c) 验证 Loose Range 证明即通过使用子协议 VeriLoosePf 的 C_2 验证 x_2 是足够小的；
- d) 如果所有 3 个步骤都得到验证,输出接受,否则拒绝。

F.1.5.2 子协议 VeriSqrtPf

该协议通过 $C_1 = a_1^{x_3} g^{r_1}$ 验证 $a_1^{x_3}$ 的二次方根的知识证明。

该证明由 (U_1, c, y_1, u_1, s_1) 组成。

算法(验证):

- a) 计算 $U_2 = a_1^{y_1} g^{s_1} U_1^{-c}$ ；
- b) 计算 $C'_2 = U_1^{y_1} g^{u_2} C^{-c}$ ；
- c) 验证是否 $= H(a_1, g, C_1, U_1, C'_2, U_2)$ 成立；
- d) 如果验证成立,输出接受。否则输出拒绝。

F.1.5.3 子协议 VeriLoosePf

该协议验证证明即满足 $C_2 = a_1^{x_2} g^{-r_1}$ 的 x_2 是足够小的。

该证明由 (c, y_1, u_1) 组成。

算法(验证):

- a) 验证 y_1 的比特长度是足够短的；
- b) 计算 $C' = a_1^{y_1} g^{u_1} C^{-c}$ ；
- c) 验证 $c = H(a_1, g, C_2, C')$ 成立；
- d) 如果验证通过,输出接受。否则输出拒绝。

F.2 证明群组成员发布协议步骤 f) 的生成正确

F.2.1 概述

机制 5 中群组成员发布协议步骤 f) 描述如下:

群组成员计算 $x = (x' + x') \bmod 2^r$ 和 $(A', h) = (a_1^x \bmod n, [x]_g)$ 。群组成员将它和一个正确生成的证明一起发送。

该证明由声称的 A' 的指数组成,对应于基 a_1 的 A' 的指数和对应于基 g 的 h 的指数是相等的。这可以使用下列证明协议。步骤 g) 对应于验证协议。

F.2.2 证据

已知 $A' = a_1^x$ 和 $h = [x]_g$ 和比特长度 $\Gamma = K_n + K + x$ 的 K_s , 安全参数 K_c , 基 a_1' 和 g 算法(证明):

- a) 随机选取一个比特长度为 Γ 正整数 x_1 ；
- b) 计算 $A_1 = a_1^{x_1}$ 和 $g_1 = [x_1]_g$ ；
- c) 计算 $c = H\{0, 1\}^{K_c}(a_1, A'; A_1, g_1)$ ；
- d) 计算 $x_2 = cx + x_1$ ；
- e) 输出 $(c; x_2)$ 。

F.2.3 验证

已知 $A' = a_1^x$, $h = [x]_g$ 和证明 $(c; x_2)$, 比特长度 $\Gamma = K_n + K + x$ 的 K_s , 安全参数 K_c , 基 a_1' 和 g

算法(验证):

- a) 验证整数 x_2 的长度和 Γ 的长度相同;
- b) 验证下列等式是否成立: $c = H\{0,1\}^{KC}(a_1, A'; a_1^{x_2}/A_1^c, [x_2]g - ch)$;
- c) 如果两个都验证通过,那么输出有效。否则输出无效。

参 考 文 献

- [1] ISO/IEC 8825-1 Information technology—ASN.1 encoding rules; Specification of basic encoding rules (BER), canonical encoding rules (CER) and distinguished encoding rules (DER)
- [2] Boneh, D., “The decision Diffie-Hellman problem”, Proc. of 3rd Algorithmic Number Theory Symposium (LNCS 1423), pp.48-63, 1998.
- [3] Brickell, E., Camenisch, J., Chen L., “The DAA scheme in context”, In Mitchell, C. (eds.) Trusted Computing. The Institute of Electrical Engineers, 2005.
- [4] Brickell, E., Camenisch, J., Chen L., “Direct anonymous attestation”, Proc. of 11th ACM Conference on Computer and Communications Security, pp.132-145, 2005.
- [5] Brickell, E., Chen, L., Li, J., “A static Diffie-Hellman attack on several direct anonymous attestation schemes”, Proc. of 4th International Conference on Trusted Systems, pp.95-111, 2012.
- [6] Brickell, E., Li, J., “A pairing-based DAA scheme further reducing TPM resources”, Proc. of 3rd International Conference on Trust and Trustworthy Computing (LNCS 6101), pp.181-195, 2010.
- [7] Brickell, E., Li, J., “Enhanced privacy ID from bilinear pairing for hardware authentication and attestation”, Proc. of 2nd IEEE Conference on Information Privacy, Security, Risk and Trust (PASSAT), pp.768-775, 2010.
- [8] Brown, D., Gallant, R., “The static Diffie-Hellman problem”, Cryptology ePrint Archive; Report 2004/306, 2004.
- [9] Canard, S., Schoenmakers, B., Stam, M., Traoré, J., “List signature schemes”, Discrete Applied Mathematics, 154(2), pp.189—201, 2006.
- [10] Chen, L., Li, J., “Revocation of direct anonymous attestation”, Proc. of 2nd International Conference on Trusted Systems, pp.128-147, 2010.
- [11] Chen, L., Page, D., Smart N., “On the design and implementation of an efficient DAA scheme”, Proc. of the 9th Smart Card Research and Advanced Application IFIP Conference, pp.223-237, 2010.
- [12] Cheon, J., “Security analysis of the strong Diffie-Hellman problem”, In Advances in Cryptology—EUROCRYPT’06, (LNCS 4004), pp.1-11, 2006.
- [13] Fujisaki, E., Okamoto, T., “Statistical zero knowledge protocols to prove modular polynomial relations”, In Advances in Cryptology—CRYPTO’97 (LNCS 1294), pp.16-30, 1997.
- [14] Furukawa, J., Imai, H., “An efficient group signature scheme from bilinear maps”, IEICE Transactions, 89-A(5), pp.1328-1338, 2006.
- [15] Hwang, J., Lee, S., Chung, B., Cho, H., Nyang, D., “Short group signatures with controllable linkability”, Proc. of the 2011 Workshop on Lightweight Security & Privacy: Devices, Protocols, and Applications, pp.44-52, 2011.
- [16] Hwang, J., Lee, S., Chung B., Cho, H., Nyang, D., “Group signatures with controllable linkability for dynamic membership”, Information Sciences, vol.222, pp.761-778, 2013.
- [17] Isshiki, T., Mori, K., Sako, K., Teranishi, I., Yonezawa, S., “Using group signatures for identity management and its implementation”, Proc. of the 2006 Workshop on Digital Identity Management, pp.73-78, 2006.
- [18] Lysyanskaya, A., Rivest, R., Sahai, A., Wolf, S., “Pseudonym systems”, Proc. of Selected Areas in Cryptography (LNCS 1758), pp.184-199, 1999.

[19] Smyth, B., Ryan, M., Chen, L., “Direct Anonymous Attestation (DAA): ensuring privacy with corrupt administrators”. Proc. of 4th European Workshop on Security and Privacy in Ad hoc and Sensor Networks (LNCS 4572), pp.218-231, 2007.
