ICS 35.030
CCS L 80

中华人民共和国国家标准

GB/T 17902.3—202X/ISO/IEC 14888-3:2018

信息技术　安全技术　带附录的数字签名 第3部分：基于离散对数的机制

Information technology—Security techniques—Digital signatures with appendix—Part 3: Discrete logarithm based mechanisms

（ISO/IEC 14888-3:2018，IDT）

（工作组讨论稿 10.22）

在提交反馈意见时，请将您知道的相关专利连同支持性文件一并附上。

XXXX－XX－XX 发布　　XXXX－XX－XX 实施

国家市场监督管理总局
国家标准化管理委员会　发布

目　　次

前　　言 7
引　　言 8
1 范围 10
2 规范性引用文件 10
3 术语和定义 10
4 符号和缩略语 11
5 通用模型 13
5.1 参数生成过程 13
5.1.1 基于证书的机制 13
5.1.2 基于身份的机制 13
5.1.3 参数选择 14
5.1.4 域参数和验证密钥的有效性 15
5.2 签名过程 15
5.2.1 概述 15
5.2.2 生成随机化值 16
5.2.3 生成预签名 16
5.2.4 待签名消息预处理 16
5.2.5 计算见证值（签名第一分量） 16
5.2.6 计算赋值 16
5.2.7 计算签名第二分量 16
5.2.8 构造附录 17
5.2.9 构造已签名消息 17
5.3 验证过程 17
5.3.1 概述 17
5.3.2 提取见证值 18
5.3.3 验证消息预处理 18
5.3.4 提取赋值 18
5.3.5 重新计算预签名 18
5.3.6 重新计算见证值 18
5.3.7 见证值校验 18
6 基于证书的机制 19
6.1 概述 19
6.2 DSA 19
6.2.1 概述 19
6.2.2 参数 20
6.2.3 签名密钥和验证密钥的生成 20
6.2.4 签名过程 21
6.2.5 验证过程 21
6.3 KCDSA 22

6.3.1 概述 22
6.3.2 参数 23
6.3.3 签名密钥和验证密钥的生成 23
6.3.4 签名过程 23
6.3.5 验证过程 24
6.4 Pointcheval/Vaudenay 算法 25
6.4.1 概述 25
6.4.2 参数 25
6.4.3 签名密钥和验证密钥的生成 25
6.4.4 签名过程 25
6.4.5 验证过程 26
6.5 SDSA 27
6.5.1 概述 27
6.5.2 参数 27
6.5.3 签名密钥和验证密钥的生成 27
6.5.4 签名过程 27
6.5.5 验证过程 28
6.6 EC-DSA 29
6.6.1 概述 29
6.6.2 参数 29
6.6.3 签名密钥和验证密钥的生成 29
6.6.4 签名过程 29
6.6.5 验证过程 30
6.7 EC-KCDSA 31
6.7.1 概述 31
6.7.2 参数 31
6.7.3 签名密钥和验证密钥的生成 31
6.7.4 签名过程 32
6.7.5 验证过程 32
6.7.6 重新计算见证值 33
6.8 EC-GDSA 33
6.8.1 概述 33
6.8.2 参数 34
6.8.3 签名密钥和验证密钥的生成 34
6.8.4 签名过程 34
6.8.5 验证过程 34
6.9 EC-RDSA 35
6.9.1 通用要求 35
6.9.2 参数 35
6.9.3 签名密钥和验证密钥的生成 36
6.9.4 签名过程 36
6.9.5 验证过程 37
6.10 EC-SDSA 37
6.10.1 通用要求 37
6.10.2 参数 38

6.10.3 签名密钥和验证密钥的生成 38
6.10.4 签名过程 38
6.10.5 验证过程 39
6.11 EC-FSDSA 39
6.11.1 通用要求 39
6.11.2 参数 40
6.11.3 签名密钥和验证密钥的生成 40
6.11.4 签名过程 40
6.11.5 验证过程 41
6.12 SM2 41
6.12.1 通用要求 41
6.12.2 参数 42
6.12.3 签名密钥和验证密钥的生成 42
6.12.4 签名过程 42
6.12.5 验证过程 43
7 基于身份的机制 44
7.1 通用要求 44
7.2 IBS-1 44
7.2.1 通用要求 44
7.2.2 参数 45
7.2.3 主密钥和签名/验证密钥的生成 45
7.2.4 签名过程 45
7.2.5 验证过程 46
7.3 IBS-2 46
7.3.1 概述 47
7.3.2 参数 47
7.3.3 主密钥和签名/验证密钥的生成 47
7.3.4 签名过程 47
7.3.5 验证过程 48
7.4 Chinese IBS 48
7.4.1 概述 48
7.4.2 参数 49
7.4.3 主密钥和签名/验证密钥的生成 49
7.4.4 签名过程 49
7.4.5 验证过程 50
附 录 A 52
（规范性） 52
对象标识符 52
附 录 B 55
（规范性） 55
转换函数(I) 55
B.1 从域元素到整数的转换：$\mathrm{FE2I}(r,x)$ 55
B.2 从整数到域元素的转换：$\mathrm{I2FE}(r,x)$ 55

B.3 从域元素到二进制字符串的转换：FE2BS(r,x) 56
B.4 从二进制字符串到整数的转换：BS2I(g,x) 56
B.5 从整数到二进制字符串的转换：I2BS(g,x) 56
B.6 整数与八位位组字符串之间的转换：I2OS(h,x)和 OS2I(h,M) 57
I2OS(h,x)： 57
输入值： 57
附 录 C 58
（资料性） 58
转换函数（II） 58
附 录 D 60
（规范性） 60
DSA 域参数的生成 60
D.1 素数p和q的生成 60
D.2 生成元 G 的生成 60
D.2.1 不可验证的 G 生成 60
D.2.2 可验证的 G 生成 60
附 录 E 62
（资料性） 62
Weil 配对与 Tate 配对 62
E.1 概述 62
E.2 函数f、g和d 62
E.3 Weil 配对 63
E.4 Tate 配对 63
E.5 约化 Tate 配对 63
附 录 F （资料性） 数值示例 64
附 录 G （资料性） 签名方案比较 127
附 录 H （资料性） 选择机制的特征声明 129
参 考 文 献 130

前　言

本文件按照GB/T 1.1—2020《标准化工作导则　第1部分：标准化文件的结构和起草规则》的规定起草。

本文件是GB/T 17902《信息技术　安全技术　带附录的数字签名》的第3部分。GB/T 17902已经发布了以下部分：

——第1部分：概述；

——第2部分：基于整数因子分解的机制；

——第3部分：基于离散对数的机制。

本文件代替GB/T 17902.3—2005《信息技术　安全技术　带附录的数字签名　第3部分：基于证书的机制》，与GB/T 17902.3—2005相比，除结构调整和编辑性改动外，主要技术变化如下：

- 调整了标准名称和内容结构，由基于证书的机制调整为基于离散对数的机制；
- 增加了 KCDSA、SDSA 以及多类椭圆曲线签名机制，并增加了 SM2 和中国 IBS 等机制（见第 6 章和第 7 章）；
- 增加了参数选择、安全等级、杂凑函数选择、域参数与验证密钥有效性确认等通用技术要求（见 5.1.3、5.1.4）；
- 增加和完善了对象标识符、转换函数、DSA 域参数生成和数值示例等内容（见附录 A、附录 B、附录 D 和附录 F）。

本文件等同采用ISO/IEC 14888-3:2018《信息技术 安全技术　带附录的数字签名　第3部分：基于离散对数的机制》，文件类型由ISO的技术规范调整为我国的国家标准。

本文件做了下列最小限度的编辑性改动：

——为与现有标准协调，将标准名称改为《信息技术　安全技术　带附录的数字签名　第3部分:基于离散对数的机制》；

——用小数符号“.”代替“，”；

——改正印刷错误；

——删除多语种出版的ISO/IEC标准化文件版本中某些语种；

——用“本文件”代替ISO/IEC标准化文件提及自身时的表述；

——删除ISO/IEC标准化文件的封面、目次、前言和引言；

——要素“规范性引用文件”文件清单中，用等同采用国际标准的GB/T 17902.1代替了ISO/IEC 14888-1，用GB/T 18238.3代替了ISO/IEC 10118-3。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本文件由全国网络安全标准化技术委员会（SAC/TC260）提出并归口。

本文件起草单位：中国电子技术标准化研究院、中国科学院信息工程研究所、中国科学院软件研究所、湖北省数字证书认证管理中心有限公司、北京信安世纪科技股份有限公司、北京时代亿信科技股份有限公司、中国电子科技集团公司第十五研究所、大唐微电子技术有限公司、中国电子信息产业集团有限公司第六研究所、河南省信息化集团有限公司、广西电网有限责任公司、北京数字认证股份有限公司、国家工业信息安全发展研究中心、西安电子科技大学。

本文件主要起草人：王姣、刘丽敏、张博然、胡可欣、王寒生、陈诚、、唐鸿宇、刘伟丰、霍珊珊、刘健、、王丹丹、雷翻翻、牛瑞彬、王龙、王子涛、李超、曾明霏、王震、赵松、赵冉、倪骏、朱辉。

本文件及其所代替文件的历次版本发布情况为：

——2005年首次发布为GB/T 17902.3—2005；

——本次为第一次修订。

引　言

数字签名机制是一类非对称密码机制，被广泛用于实体鉴别、数据来源鉴别、数据完整性和抗抵赖服务。有两种数字签名机制：

——若在验证过程中，需要消息作为输入的一部分，则此类机制称为“带附录的数字签名”，附录计算需要使用杂凑函数；

——若在验证过程中，披露全部或是部分消息，则此类机制称为“带消息恢复的数字签名”，签名生成和验证也会使用到杂凑函数。

带附录的数字签名在 GB/T 17902 中进行了规范，带消息恢复的签名机制（ISO/IEC 9796 系列规范了此类机制）；杂凑函数由 ISO/IEC 10118 系列规范。

GB/T 17902《信息技术　安全技术　带附录的数字签名》旨在规定带有附录的数字签名机制，拟由三个部分构成。

——第1部分：概述。目的在于规范通用的带附录数字签名的整体框架和通用模型。

——第2部分：基于整数因子分解的机制。目的在于规范基于整数因子分解的数字签名机制。

——第3部分：基于离散对数的机制。目的在于规范基于离散对数的数字签名机制。

本文件为GB/T 17902的第3部分。

注 1：在某些应用中，允许在不知道签名密钥的情况下对先前已签名的消息产生新的签名。此类应用的一个示例是 ISO/IEC 20008 中规定的匿名数字签名机制中的成员凭证。

数字签名机制基于非对称密码技术，并涉及以下三个基本操作：

——用于生成密钥对的过程，其中每个密钥对由一个私有签名密钥和相应的公开验证密钥；

——使用签名密钥的过程，称为签名过程；

——使用验证密钥的过程，称为验证过程。

以下是两种类型的数字签名机制：

——当对于给定的签名密钥，对同一消息产生的任何两个签名总是相同时，该机制称为确定性的（或非随机化的）（详见 GB/T 17902.1）；

——当对于给定的消息和签名密钥，签名过程的任何两次应用产生（极大概率）两个不同的签名，该机制被称为随机化（或非确定性）机制。

本文件规定的机制均为随机化机制。

数字签名机制还能分为以下两类：

——当整个消息必须与签名一起存储和/或传输时， 该机制被称为“带附录的数字签名机制”（此类机制是 GB/T 17902 系列的主题）；

——当整个消息或其部分能从签名中恢复时，该机制称为具有消息恢复功能的签名机制（ISO/IEC 9796 系列规定了此类别中的机制）。

数字签名的验证需要访问签名实体的验证密钥。因此，验证方能够将正确的验证密钥副本与签名实体相关联，或更一般地，验证密钥与签名实体身份数据（或其部分）之间的关联非常重要。身份数据与验证密钥之间的关联可由外部实体或机制来保证，或以某种方式内置于验证密钥本身。前一种情况称为“基于证书的”，后一种情况称为“基于身份的”。通常，在基于身份的方案中，验证方能根据签名者的身份数据计算出签名者的验证密钥。本文件规定的数字签名机制分为基于证书的机制和基于身份的机制。

数字签名机制的安全性基于一个难解的计算问题，即根据当前知识，找到解决方案在计算上不可行的问题，例如分解问题和离散对数问题。本文件规定了基于离散对数问题的带附录的数字签名机制，而 GB/T 17902.2 规定了基于因子分解问题的带附录的数字签名机制。

本文件包含了 14 种机制：DSA、Pointcheval/Vaudenay 算法、EC-DSA、EC-KCDSA、EC-GDSA、KCDSA、IBS-1、IBS-2、SRA、EC-RDSA、EC-SDSA、EC-FSDSA、SM2 和 SM9。本文件规定的机制采用抗碰撞杂凑函数对要签名的消息（可能包含多个部分）进行杂凑运算。ISO/IEC 10118 规定了杂凑函数。

安全技术　带附录的数字签名
第 3 部分：基于离散对数的机制

1　范围

本文件规定了基于离散对数问题的带附录的数字签名机制。本文件提供了：
——带附录的数字签名机制的通用描述；
——提供带附录的数字签名的多种机制。
对于每种机制，本文件规定了：
——生成密钥对的过程，
——生成签名的过程，
——验证签名的过程。
附录A定义了分配给本文件规定的数字签名机制的对象标识符，并定义了算法参数结构。
附录B定义了本文件中使用的FE2I、I2FE、FE2BS、BS2I、I2BS、I2OS和OS2I转换函数。
附录D定义了如何生成DSA域参数。

2　规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；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 17902.1—2023　信息技术　安全技术　带附录的数字签名　第1部分：概述（ISO/IEC 14888-1:2008，IDT）

GB/T 18238.3—2024　网络安全技术　杂凑函数　第3部分：专门设计的杂凑函数（ISO/IEC 10118-3:2018，NEQ）

注：GB/T 18238.3—2024被引用的内容与ISO/IEC 10118-3：2018被引用的内容没有技术上的差异。

3　术语和定义

GB/T 17902.1界定的以及下列术语和定义适用于本文件。

3.1

有限交换群　finite commutative group

具有二元运算“*”的有限集合 E，使得；
——对于所有群元素 $a, b \in E$，有 $a * b \in E$；
——对于所有群元素$a, b, c \in E, (a * b) * c = a * (b * c)$；
——存在一个群元素 e ∈ E，使得对所有 a ∈ E 都有 e * $a = a$，其中 e 称为该群的单位元；
——对于所有群元素 $a \in E$，存在一个群元素 $b \in E$，使得 $b * a = e$；
——对于所有群元素$a, b \in E, a * b = b * a$。

注1：在某些情况下，例如当E是椭圆曲线上的点集时，有限集E中的运算采用加法符号表示。

3.2

循环群 cyclic group

有限交换群 E(3.1)，具有 n 个元素，包含一个群元素$a \in E$，称为生成元，其阶为 n。

3.3

椭圆曲线群 elliptic curve group

定义在有限域上椭圆曲线点集上的循环群（3.2）。

注1：设 $F = GF(r)$表示阶为 r 的伽罗瓦域，其中 r 为奇素数 p ，或 r 等于2^m（m 为正整数）。定义在有限域F上的椭圆曲线可由仿射曲线方程确定，形式为$y^2 = x^3 + a_1x + a_2$（当$r = p$，其中p为某个奇素数时）或$y^2 + xy = x^3 + a_1x^2 + a_2$（当$r = 2^m$，其中m为某个正整数时），其中系数$a_1$和$a_2$是域F中适当选取的元素。对应的椭圆曲线$E$包含$F \times F$上的一组仿射点，以及一个特殊的非仿射点“无穷远点”；椭圆曲线E上的仿射点P可表示为有序对$(Px, Py) \in F \times F$：若令$x = P_x$，$y = P_y$且全部运算在域 F 中进行，则该有序对满足给定的仿射曲线方程；令“+”表示称为“椭圆曲线加法”的二元运算，对E上的（大多数）仿射点，该运算由经典弦切法则定义。在E的仿射点集合增加0_E（一个特殊的“无穷远”点，作为“+”的单位元，但不能表示为有序对）之后，集合E连同二元运算“+”构成一个有限交换椭圆曲线群E。

注 2：椭圆曲线群 E 的阶等于满足椭圆曲线仿射方程的$F \times F$有序对个数加一。

3.4

（群元素 a 的）阶 order(of a group element a)

使$a^n = e$的最小正整数 n，其中 e 是群的单位元，a^n递归定义为$a^0 = e$，且当 m>0 时 $a^m = a * a^{m-1}$，$*$代表群运算。

3.5

双线性对 pairing

一个函数，其输入为来自有限域上椭圆曲线群（3.3）G_1的两个元素 P 和 Q，输出为来自有限域上另一循环群（3.2）G_2的一个元素，并具有以下两个性质（假设循环群G_1和G_2的阶为 q，其中 q 为素数，对于任意两个元素 P、Q，配对函数的输出记为$<P,Q>$）：

——双线性：若 P、P_1、P_2、Q、Q_1、Q_2是 G_1的元素，且 a 是满足 $1 \le a \le q-1$ 的整数，则：

- $<P_1 + P_2, Q> = <P_1, Q> * <P_2, Q>$，
- $<P, Q1 + Q2> = <P, Q1> * <P, Q2>$，且
- $<[a]P, Q> = <P, [a]Q> = <P, Q>^a$ ；

——非退化性：如果P是G_1的非单位元，则$<P, P> \neq 1$。

3.6

可信密钥生成中心 trusted key generation centre KGC

可信第三方，在基于身份的签名机制中，为每个签名实体生成签名密钥。

4 符号和缩略语

下列符号与缩略语适用于本文件。

$a \oplus b$　a和b的按位异或，其中a和b可为比特或者等长比特串；若为比特串，则按比特执行异或运算

a_1, a_2　椭圆曲线系数

$a \bmod n$　对于任意整数a和正整数n，存在某整数b，满足$r = a - bn$，且$0 \leq r \leq$（$n - 1$），该唯一整数余数r即为$a \bmod n$。

(A, B, C)　签名公式系数，对于第 6 章规定的机制，该组系数定义签名的计算方式

注 1：签名公式在 5.2.1 中规定。

D　用于规定签名密钥与验证密钥之间关系的参数

E　由两个椭圆曲线系数a_1和a_2定义的椭圆曲线

E　有限交换群；对于基于乘法群的机制，E的元素属于Z_p^*；对于基于椭圆曲线点加法群的机制，E的元素是定义在$GF(r)$上椭圆曲线E上的点

$\#E$　E的阶；对于基于乘法群Z_p^*的机制，$\#E$等于$p - 1$；对于基于椭圆曲线点加法群的机制，$\#E$为 $\mathrm{GF}(r)$上椭圆曲线 E 上的点数加 1[包括0_E（无穷远点）]

F　有限域

F_p　阶为 p 的有限域

$\gcd(N_1, N_2)$　整数N_1和N_2的最大公约数

G　E 中阶为 q 的元素

$\mathrm{GF}(r)$　阶为 r 的有限域，其中 r 为素数幂

G_1　素数阶q的循环群；G_1中的元素是$GF(r)$上椭圆曲线上的点

G_2　素数阶q的循环群；G_2中的元素是有限域$GF(r)$中的元素

H_1　将数据字符串转换为G_1中元素的杂凑函数

注 2：输入的数据字符串首先转换为整数，再通过附录 C 规定的整数转点函数（I2P 函数），将该整数转换为 $\mathrm{GF}(r)$上椭圆曲线E上的点

h, H_2　杂凑函数，即在 GB/T 18238.3 中规定的机制之一

ID　包含签名者标识符的数据串，用于 SM2、IBS-1、IBS-2 和中国 IBS 机制

m　嵌入次数（或扩张次数）

$[n]P$　乘法运算，输入正整数n和曲线E上的点P，输出曲线E上的另一个点Q，其中$Q = [n]P = P + P + \ldots + P$（相加$n - 1$次）。该运算满足$[0]P = 0_E$（无穷远点），且$[-n]P = [n](-P)$。

P　G_1的生成元，用于 IBS-1、IBS-2 和中国 IBS 机制

p　素数或素数的幂

q　素数，为$\#E$的因子，且为G_1和G_2的阶

r　$\mathrm{GF}(r)$的阶；在基于椭圆曲线点加法群的机制中，r为素数幂p^m，其中素数$p \geq 2$，整数$m \geq 1$。

T　赋值

T_1　赋值T的第一部分

T_2　赋值T的第二部分

U　KGC 的主私钥，为随机选取生成的整数，用于 IBS-1、IBS-2 和中国 IBS 机制

V　KGC 的主公钥，G_1的一个元素，用于 IBS-1、IBS-2 和中国 IBS 机制

Z_N^*　满足 $0 < i < N$且 $\gcd(i, N) = 1$ 的整数集合，算术运算模N定义

Z_p^*　满足 $0 < i < p$且p为素数的整数集合，是一个乘法群

α	素数（或素数的幂）p的比特长度
β	素数q的比特长度
γ	杂凑函数h和H_2的输出比特长度
Π	预签名
Π_X	是Π的x坐标，其中$\Pi = (\Pi_X, \Pi_Y)$是椭圆曲线上的点
Π_Y	是Π的y坐标，其中$\Pi = (\Pi_X, \Pi_Y)$是椭圆曲线上的点
Π_a	是Π的第一个元素，其中$\Pi = (\Pi_a, \Pi_b)$是二次扩域的元素
Π_b	是Π的第二个元素，其中$\Pi = (\Pi_a, \Pi_b)$是二次扩域的元素
0_E	是椭圆曲线E上的无穷远点
<>	双线性非退化对。
||	$X\|\|Y$用于表示数据项X和Y按指定顺序拼接的结果。

5 通用模型

5.1 参数生成过程

5.1.1 基于证书的机制

5.1.1.1 域参数的生成

对于基于离散对数的数字签名机制，域参数集包括以下参数：

——E,有限交换群；

——q,#E 的素因子；

——G,群E中阶为q的元素。

群E使用乘法记号。值得注意的是，所选特定签名机制可能对E、q和G的选取施加额外约束。

5.1.1.2 签名密钥和验证密钥的生成

签名实体的签名密钥为秘密生成的随机或伪随机整数X，满足 $0 < X < q$。对应的验证密钥Y是E上的元素，按公式(1)计算：

$$Y = G^{X^D} \qquad (1)$$

其中D是由所用签名机制定义的参数。D的取值只能为-1 或 1。

注：若实现排除了少数整数作为可能的X值，仍视为符合规范。例如，可排除值 1，因为该值会导致用户的验证密钥成为生成元G，该情形极易被检测识别。

5.1.2 基于身份的机制

5.1.2.1 符号表示

第 7 章中规定的两种基于身份的机制均基于椭圆曲线群上的双线性对实现。为描述基于身份的机制，采用加法群记法。

5.1.2.2 域参数的生成

域参数集包括以下参数：

——E，有限交换群；

——$GF(r)$，阶为 r 的伽罗瓦域

——G_1，阶为素数q的循环群；

——G_2，阶为素数q的循环群；
——P，G_1的一个生成元；
——q，素数——G_1和G_2的阶；
——<>双线性非退化对。

5.1.2.3 主密钥的生成

KGC 的主私钥是秘密生成的随机或伪随机整数U，满足 0< U < q。对应的主公钥 V 是G_1中的一个元素，按如下公式计算：

$$V = [U]P$$

5.1.2.4 签名密钥和验证密钥的生成

签名实体的签名密钥是G_1中的一个元素，由 KGC 按公式(2)计算：

$$X = [U]Y \quad (2)$$

其中：

U——KGC 的主私钥；

Y——$Y = H_1(ID)$，是验证密钥。

其中：

ID——KGC 的身份字符串；

H_1——一个杂凑函数。

5.1.3 参数选择

5.1.3.1 参数长度选择

典型安全等级的参数位长度如表 1 所示。建议的最低安全等级为2^{112}。

注 1：安全等级是指针对密码原语的已知最优攻击所需执行的运算步数。如果对杂凑函数的已知最优攻击需要2^{112}步，则该杂凑函数的安全等级为2^{112}。关于参数长度的全面分析参见参考文献[25]和[34]。

选取参数α、β和γ时，不必要求三者具备相同安全等级；已实现的签名方案的安全等级取各参数安全等级的最小值。

表 1 不同安全等级的参数长度

安全等级	2^{80}	2^{112}	2^{128}	2^{192}	2^{256}
α	1 024	2 048	3 072	7 680	15 360
β	160	224	256	384	512
γ	160	224	256	384	512

建议安全级别2^{80}仅用于遗留应用。

注 2：本文件规定的机制并非全部支持本表所列的全部安全等级。例如，6.1 中的 DSA 仅支持最高2^{128}的安全等级。

5.1.3.2 杂凑函数的选取

杂凑函数的选择应基于 GB/T 18238.3 中标准化的杂凑函数。即，h和H_2应为 GB/T 18238.3 中规定的机制之一，而H_1将使用 GB/T 18238.3 中规定的机制之一获得的数据字符串转换为G_1中的元素。

本文件使用的杂凑函数宜具有抗碰撞性。本文件中所选杂凑函数的安全强度宜满足或超

过密钥生成参数的安全强度。杂凑函数安全级别与密钥生成参数之间的关系见 5.1.3.1。

此外，验证数字签名的实现应具有安全确定签名者使用何种杂凑函数的方法。否则，攻击者可能能够说服验证方使用更弱的杂凑函数，从而绕过预期的安全级别。

5.1.4 域参数和验证密钥的有效性

签名验证方可能需要确保域参数和验证密钥有效，否则即使签名验证通过，也无法确保达到预期安全性，且攻击者可能能够生成可验证的签名。可通过以下方式之一确保域参数的有效性：

——从可信的公开来源（如标准）中选择有效的域参数；

——三方（如 CA 或 KGC）生成有效的域参数；

——由可信第三方（如 CA 或 KGC）验证候选域参数；

——对签名者而言，由签名者使用可信系统生成有效的域参数；

——由用户（即签名者或验证方）验证候选域参数。

可通过以下方式之一确保验证密钥的有效性：

——对于签名者，使用可信系统生成公开密钥/私有签名密钥对；

——对于签名者或验证方，由可信第三方（如 CA 或 KGC）对公开验证密钥进行有效性验证；

——用户（即签名者或验证方）对验证密钥的验证。

注 1：需要对域参数和密钥进行验证。然而，如何实现这一点不在本文件范围之内。

注 2：对签名者进行身份验证的方法取决于实际应用，这不在本文件范围之内。

5.2 签名过程

5.2.1 概述

本文件中的所有签名机制均使用随机化值K。该随机值（与消息共同）生成见证值R（签名第一分量）以及一组赋值(T_1, T_2)。消息的签名为二元组(R, S)，其中S（签名第二分量）通过求解签名公式得到。

在第 6 章规定的基于证书的机制中，签名公式为公式(3)：

$$AK + BX^D + C \equiv 0(\bmod q) \tag{3}$$

其中：

(A, B, C)——(S, T_1, T_2)或$(S, T_1, S + T_2)$的一个排列；

X ——签名私钥；

D ——依赖于特定机制的参数。

在第 7 章规定的基于身份的机制中，签名公式为公式(4)：

$$[K]A + [U^D]B + C \equiv 0_E(\text{in } G_1) \tag{4}$$

其中：

(A, B, C)——(S, T_1, T_2)或$(S, T_1, [Y^{-1}]S + T_2)$的一个排列；

U ——主私钥；

Y ——公共验证密钥；

D ——依赖于特定机制的参数。

该排列将在签名系统初始化时指定或协商确定。

签名流程以及带签名消息的形成包括以下八个阶段（见图 1）：

——生成随机化值；

——生成预签名；

——待签名消息预处理；

——计算见证值；
——计算赋值（基于身份的机制无需计算该赋值）；
——计算签名第二分量；
——构造附录；
——构建签名消息。
在此过程中，签名实体使用其签名私钥、公共验证密钥（可选）以及域参数。

5.2.2 生成随机化值

对每一次签名，签名实体都重新生成一个秘密随机化值，即整数K，满足$0 < K < q$。该阶段的输出为K，应保密并在使用后安全销毁。

注 1：随机化值K可视为临时密钥。

注 2：与 5.1.1.2 同理，实现时可排除少量整数作为K的可选值，不影响合规性。

5.2.3 生成预签名

本阶段输入随机化值K，以及可选的签名私钥X；签名实体结合K和公开参数计算预签名Π。在第 6 章规定的基于证书的机制中，按公式(5)在群E中计算预签名：

$$\Pi = G^K \tag{5}$$

在E中，第 7 章基于身份机制的预签名计算方式在对应子机制中单独规定。本阶段输出为预签名Π。

5.2.4 待签名消息预处理

在消息预处理过程中，将M_1和M_2中的一个置为消息M（可选地添加前缀），另一个置空。

5.2.5 计算见证值（签名第一分量）

本阶段输入取自 5.2.3 节输出的预签名Π以及 5.2.4 节输出的M_1；将二者作为见证函数的输入，见证函数输出见证值R。见证函数由各具体签名机制规定。

5.2.6 计算赋值

赋值函数的输入为签名第一分量（5.2.5 节的见证值R）、5.2.4 节得到的M_2，以及可选的验证密钥Y。赋值函数输出赋值$T = (T_1, T_2)$。在第 6 章规定的基于证书的机制中，T_1、T_2为整数，满足：

$$0 < |T_1| < q, 0 < |T_2| < q$$

在第 7 章规定的基于身份的机制中，T_1、T_2为群G_1中的元素。在基于身份的机制中，无需计算赋值T。

5.2.7 计算签名第二分量

本阶段输入 5.2.1 节的随机化值K、签名密钥X、5.2.6 节的赋值$T = (T_1, T_2)$、(S, T_1, T_2)的排列(A, B, C)、5.1.1.2 节定义的变量D以及 5.1.1.1 和 5.1.2.1 节中定义的域参数q。

在基于证书的机制中，签名实体构建签名公式（见公式(6)）：

$$AK + BX^D + C \equiv 0 (\bmod q) \tag{6}$$

求解签名公式得到签名第二分量S，满足$0 < S < q$。

在基于身份的机制中，签名实体求解签名公式，得到属于群G_1的签名第二分量S，使得$S \in G_1$。该解满足签名公式（见公式(7)）：

$$[K]A + [U^D]B + C \equiv 0_E(\text{in } G_1) \tag{7}$$

二元组(R,S)称为签名，记为Σ。

5.2.8 构造附录

附录由签名与可选文本域$text$构成，即$[(R,S),text]$。文本域可存放证书，通过密码学手段将公共验证密钥与签名实体的身份信息进行绑定。如 GB/T 17902.1 所述，不同应用场景下附录的构造与拼接方式各不相同。通用要求是验证方能够将签名与对应消息正确关联。为了保证签名验证成功，在签名验证之前，验证方能够将正确的验证密钥与签名建立对应关系。

5.2.9 构造已签名消息

将消息M与附录拼接，得到带签名消息，即$M|||[(R,S),text]$。

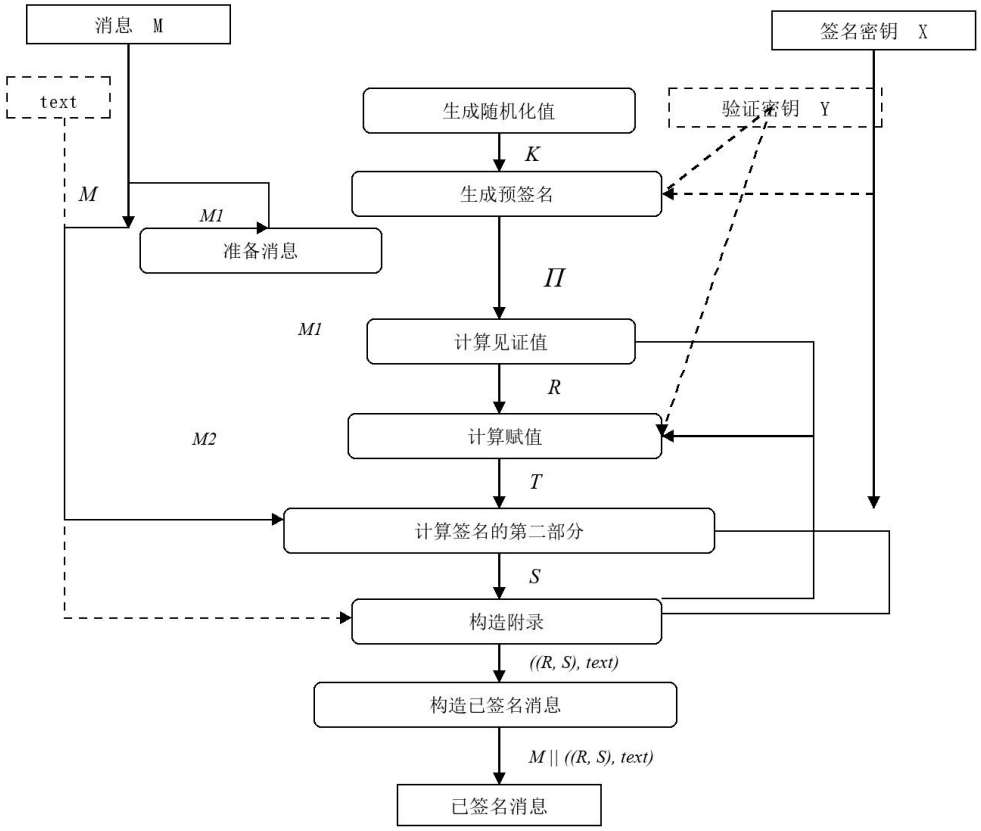


图 1 带随机见证值的签名过程（将M_1和M_2中的一个置为消息M，另一个置空）

5.3 验证过程

5.3.1 概述

验证过程包括以下 6 个阶段（见图 2）：

——提取见证值；

——验签消息预处理；

——提取赋值（在基于身份签名机制中赋值为可选计算项）；

——重新计算预签名；

——重新计算见证值；

——见证值校验。

在此过程中，验证方使用签名者的验证密钥、KGC（密钥生成中心）的主公钥（仅适用于第 7 章中规定的基于身份的机制）以及域参数。

5.3.2 提取见证值

验证方从附录中解析出签名(R,S)，分离得到见证值R和签名第二分量S。同时，验证方根据对应签名机制制定的规则，验证签名分量R和S的取值范围或比特长度。若不满足预定义的规则，则应拒绝该签名。

5.3.3 验证消息预处理

验证方从带签名消息中提取出原始消息M，并将消息拆分为M_1、M_2两部分。

5.3.4 提取赋值

本阶段的运算逻辑与 5.2.6 节相同。赋值函数输入 5.3.2 节得到的见证值R、5.3.3 节得到的M_2以及验证密钥Y（可选）。通过赋值函数重新计算得到赋值$T=(T_1,T_2)$。在基于身份的机制中，无需重新计算T。

5.3.5 重新计算预签名

本阶段输入域参数集、验证密钥Y、5.3.4 节的赋值$T=(T_1,T_2)$、5.3.2 节得到的签名第二分量S以及 5.3.2 节的R（可选）。验证方根据签名函数指定的顺序将值(S,T_1,T_2)赋给系数(A,B,C)，并在基于证书的机制中计算Π'。

在基于证书的机制中，Π'在E上按公式(8)计算：

$$\Pi' = Y^m G^n \tag{8}$$

其中

$$m = -A^{-1}B \bmod q;$$

$n = -A^{-1}C \bmod q$。

在基于身份的机制中，Π'的计算逻辑在对应子机制中单独规定。

5.3.6 重新计算见证值

本阶段的运算逻辑与 5.2.5 节相同。验证方调用见证函数，输入 5.3.5 节中得到的Π'以及 5.3.3 节中得到的M_1，输出重新计算得到的见证值R'。在 7.3 节中规定的 IBS-2 机制中不计算R'，改为计算两个验证函数。

5.3.7 见证值校验

若 5.3.6 节中得到的重新计算的见证值R'等于 5.3.2 节中的R，则签名验证通过。在 7.3 中规定的 IBS-2 机制中，不再校验等式$R=R'$，改为校验上述两个验证函数的输出值是否相等。

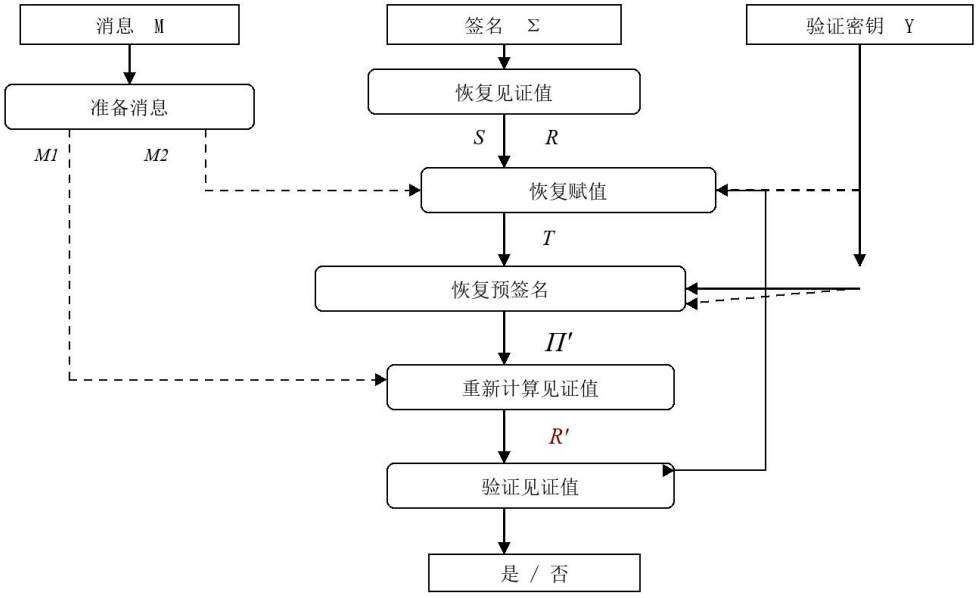


图 2　带随机见证值的验证过程

6　基于证书的机制

6.1　概述

第 6 章规定了 11 种基于证书的机制。这些机制采用乘法群$Z_p{}^*$或椭圆曲线点的加法群完成运算。采用乘法群$Z_p{}^*$算术运算的机制包括数字签名算法（DSA）、韩国基于证书的数字签名算法（KCDSA）、Pointcheval/Vaudenay 算法以及 Schnorr 数字签名算法（SDSA）。采用椭圆曲线点加法群运算的算法包括椭圆曲线 DSA（EC-DSA）、椭圆曲线 KCDSA（EC-KCDSA）、椭圆曲线德国数字签名算法（EC-GDSA）、椭圆曲线俄罗斯数字签名算法（EC-RDSA）、椭圆曲线 Schnorr 数字签名算法（EC-SDSA）、椭圆曲线完整 Schnorr 数字签名算法（EC-FSDSA）以及 SM2 算法。

在椭圆曲线运算中，椭圆曲线点采用仿射坐标表示，即椭圆曲线点Π包含两个坐标分量：x坐标Π_x，y坐标Π_y。EC-DSA、EC-KCDSA、EC-GDSA、EC-RDSA、EC-SDSA、EC-FSDSA 与 SM2 所使用的椭圆曲线限定为非奇异、非超奇异曲线。

杂凑函数标识符可用于绑定签名机制和杂凑函数。

6.2　DSA

6.2.1　概述

DSA（数字签名算法）是一种签名机制，其中$E = Z_p^*$，p为素数，q为整除$p-1$ 的素数。DSA 的参数D等于 1。消息准备使得M_1为空，M_2为待签名的消息，即$M_2 = M$。见证函数由公式(9)定义：

$$R = \Pi \bmod q \tag{9}$$

其中，对于某些整数K，$\Pi = G^K \bmod p$，

赋值函数由公式(10)定义：

$$(T_1, T_2) = (-R, -BS2I(\gamma, H)) \quad (10)$$

其中$H = h(M)$是消息M的截断杂凑码，根据附录 B 给出的转换规则转换为整数。DSA 签名公式中的系数(A, B, C)按公式(11)设置：

$$(A, B, C) = (S, T_1, T_2) \quad (11)$$

因此，签名公式变为公式(12)：

$$SK - RX - BS2I(\gamma, H) \equiv 0(\bmod q) \quad (12)$$

注：本机制取自参考文献[17]。此处符号与参考文献[17]略有不同，以符合本文件其他部分使用的符号。

6.2.2 参数

p 是一个素数，$2^{\alpha-1} < p < 2^{\alpha}$。

q 是$p-1$ 的一个质因数，$2^{\beta-1} < q < 2^{\beta}$。

G 是阶为q的子群的生成元，满足 $1 < G < p$。

DSA 中允许四种(α, β)对的选择，分别为（1 024,160）、（2 048,224）、（2 048,256）和（3 072,256）。除非参与实体之间已达成协议使用更强的杂凑函数，否则建议(α, β) 对与γ的安全强度相同。整数p、q和G可以是公开的，并且可以为一组用户所共用。

参数p、q和G按照附录 D 的规定生成。如果不需要与 NIST 联邦标准兼容，则可以使用 ISO/IEC 18032 中给出的素数生成技术来生成参数p和q。

建议所有用户按照参考文献[17]检查 DSA 公共参数是否正确生成。

注：如果签名者可以自由地特意选择域参数q以促进杂凑值之间的碰撞，则可能对此类 DSA 实例发起攻击，其中对底层杂凑函数所需的碰撞可以以2^{74}、2^{101}或2^{114}的复杂度找到（分别对应于γ=160、224 或 256），而最安全的实例中查找碰撞的复杂度为2^{80}、2^{112}或2^{128}。[39]然而，该攻击很容易被检测到。此外，当按照参考文献[17]的规定生成域参数（其中包括附录 D 规定的方法）时，无法实施攻击。如果无法验证为生成域参数使用了适当的方法，则也能通过使用 6.3、6.4 和 6.7 中规定类型的机制来防止该攻击。

建议基于 SHA-1 的数字签名仅用于遗留应用。

6.2.3 签名密钥和验证密钥的生成

签名实体的签名密钥是秘密生成的随机或伪随机整数X，满足 $0 < X < q$。参数D为 1。相应的验证密钥 Y 由公式(13)给出：

$$Y = G^X \bmod p \quad (13)$$

用户的签名密钥X和验证密钥Y通常在一段时间内固定。签名密钥X应保密。

6.2.4 签名过程

6.2.4.1 生成随机化值

签名实体计算一个随机或伪随机整数K，使得 $0 < K < q$。

6.2.4.2 生成预签名

此阶段的输入是随机化值K，签名实体根据公式(14)进行计算：

$$\Pi = G^K \bmod p \tag{14}$$

6.2.4.3 准备待签名消息

对消息进行准备，使得M_1为空，M_2为待签名的消息，即$M_2 = M$。

6.2.4.4 计算见证值

签名实体计算$R = \Pi \ mod \ q$，其中见证值仅为预签名的函数。因此，见公式(15)：

$$R = \left(G^K \bmod p\right) \bmod q \tag{15}$$

6.2.4.5 计算赋值

签名实体计算杂凑码；若所选杂凑函数的输出比特长度大于$\lceil log2q \rceil$，则将H设置为$h(M_2)$最左边（最高有效）的$\lceil log2q \rceil$特。否则，H为$h(M_2)$。随后，根据附录 B 中的转换规则 BS2I 将H转换为整数。赋值(T_1,T_2)为$(-R, -BS2I(\gamma, H))$。

6.2.4.6 计算签名的第二部分

签名为(R,S)，其中R根据 6.2.4.4 计算；并见公式(16)：

$$S = \left(K^{-1}(BS2I(\gamma, H) + XR)\right) \bmod q \tag{16}$$

应检查$R = 0$ 或$S = 0$ 是否成立。若任一条件成立，宜生成新的K值并重新计算签名（若签名生成正确，则出现$R = 0$ 或$S = 0$ 的可能性极低）。

6.2.4.7 构造附录

附录为(R,S)与可选文本字段$text$的拼接，即$((R,S), text)$。

6.2.4.8 构造已签名消息

已签名消息是消息M与附录的拼接：$M||((R,S), text)$。

6.2.5 验证过程

6.2.5.1 概述

在验证已签名消息的签名之前，验证方须拥有p、q、G和Y的可信副本。

6.2.5.2 恢复见证值

验证方从附录中检索见证值R和签名的第二部分S。验证方检查是否满足 $0 < R < q$且$0 < S < q$。如果任一条件不满足，则应拒绝该签名。

6.2.5.3 准备待验证消息

验证方从已签名消息中检索$M_2 = M$。M_1为空。

6.2.5.4 恢复赋值

此阶段与 6.2.4.5 相同。赋值函数的输入包括来自 6.2.5.2 的见证值R和来自 6.2.5.3 的M_2。赋值$T = (T_1, T_2)$作为赋值函数（6.2.4.5）的输出被重新计算。

6.2.5.5 重新计算预签名

该阶段的输入为域参数、验证密钥Y、来自 6.2.5.4 的赋值$T = (T_1, T_2)$ 以及来自 6.2.5.2 的签名第二部分S。验证方使用公式(17)获得预签名Π'的重新计算值：

$$\Pi' = Y^{-S^{-1}T_1 \bmod q} G^{-S^{-1}T_2 \bmod q} \bmod p \qquad (17)$$

6.2.5.6 重新计算见证值

此阶段的计算与 6.2.4.4 相同。验证方执行见证函数。输入为来自 6.2.5.5 的Π'。注意M_1为空。输出为重新计算的见证值R'，使得$R' = \Pi' \bmod q$。

6.2.5.7 验证见证值

验证方将来自 6.2.5.6 的重新计算见证值R'与来自 6.2.5.2 的R值进行比较。若$R' = R$，则签名通过验证。

6.3 KCDSA

6.3.1 概述

KCDSA（韩国基于证书的数字签名算法）是一种签名机制，其中$E = Z_p^*$，p为素数，q为整除$p - 1$ 的素数。验证密钥Y为$G^{x^{-1}}$；即参数D为-1。消息准备方式为M_1为空且M_2为待签名消息，即$M_2 = M$。见证函数由公式(18)定义：

$$R = h(I2BS(\alpha, \Pi)) \qquad (18)$$

若γ长于β，则见证函数替换为公式(19)：

$$R = I2BS(\beta, BS2I(\gamma, h(I2BS(\alpha, \Pi))) \bmod 2\beta) \qquad (19)$$

域参数应指明所使用的杂凑函数。赋值函数由公式(20)定义：

$$(T_1, T_2) = (V, -1) \qquad (20)$$

其中，$V = \mathrm{BS2I}(\beta, R \oplus H) mod\ q$。值$H$为来自公钥$Y$和消息$M$的杂凑码。KCDSA 签名公式的系数$(A, B, C)$按公式(21)设置：

$$(A, B, C) = (T_2, S, T_1) \qquad (21)$$

因此，签名公式变为公式(22)：

$$-K + SX^{-1} + V \equiv 0 (\bmod q) \qquad (22)$$

注：本机制取自参考文献[36]。此处记号与参考文献[36]略有不同，以符合本文件其他部分使用的记号。

6.3.2 参数

P 是一个素数，$2^{\alpha-1} < p < 2^{\alpha}$。

q 是$p-1$的一个素因子，$2^{\beta-1} < q < 2^{\beta}$。

F 对于满足 $1 < F < p-1$ 和$F^{(p-1)/q} \bmod p > 1$ 的整数F，

G $G = F^{(p-1)/q} \bmod p$是Z_p^*中阶为 q 的元素。

l 所选杂凑函数 h 的输入块大小（以比特计）。

杂凑函数标识符或带有指定杂凑函数的 OID。KCDSA 中允许使用三组(α, β, h)选择，分别为$(2\,048,224, \text{SHA}-224)$、$(3\,072,256, \text{SHA}-256)$和$(2\,048,224, \text{SHA}-256)$。其中推荐$(2\,048,224, \text{SHA}-224)$和$(3\,072,256, \text{SHA}-256)$，而在仅支持 SHA-256 而不支持 SHA-224 的情况下可使用$(2\,048,224, \text{SHA}-256)$。

整数p、q、G和l可公开，并可为一组用户所共有。

6.3.3 签名密钥和验证密钥的生成

签名实体的签名密钥是秘密生成的随机或伪随机整数X，满足 $0 < X < q$。参数D为-1。对应的验证密钥Y为公式(23)：

$$Y = G^{X^{-1}} \bmod p \tag{23}$$

用户的签名密钥X和验证密钥Y通常在一段时间内固定。签名密钥 X 应保密。

6.3.4 签名过程

6.3.4.1 生成随机化值

签名实体计算一个随机或伪随机整数K，使得 $0 < K < q$。

6.3.4.2 生成预签名

此阶段的输入是随机化值K，签名实体计算公式(24)：

$$\Pi = G^{K} \bmod p \tag{24}$$

6.3.4.3 准备待签名消息

对消息进行准备，使得M_1为空，M_2为待签名的消息，即$M_2 = M$。

6.3.4.4 计算见证值

签名实体计算见证值$R = h(I2BS(\alpha, \Pi))$，其中h的输出是从预签名Π转换的长度为α的比特串的杂凑码。如果γ比β长，则计算由$R = I2BS\left(\beta, BS2I\left(\gamma, h\left(I2BS(\alpha, \Pi)\right)\right) \bmod 2^{\beta}\right)$替换的见证值。

转换规则 I2BS 和 BS2I 在附录 B 中给出。

6.3.4.5 计算赋值

签名实体计算赋值$(T_1,T_2)=(V,-1)$，其中$V=\mathrm{BS2I}(\beta,R\oplus H)mod\ q$，$H=h(Y'||M_2)$是$Y'=\mathrm{I2BS}\left(l,Y\ mod\ 2^l\right)$与消息$M^2$拼接结果的杂凑码。$Y'$是长度为l的比特串。在计算$V$时，比特串$R\oplus H$应先转换为整数，再对$q$取模。

如γ比β长，计算$H=\mathrm{I2BS}(\beta,\mathrm{BS2I}(\gamma,h(Y'||M_2))\ mod\ 2^\beta)$。

注：Y'对于用户是一个固定值，因此该值能作为用户参数保存。

6.3.4.6 计算签名的第二部分

签名为(R,S)，其中R按6.3.4.4计算，[见公式(25)]：

$$S=X(K-V)\bmod q \tag{25}$$

6.3.4.7 构造附录

附录将由(R,S)与可选文本字段$text$的连接$((R,S),text)$组成。

6.3.4.8 构造已签名消息

已签名消息是消息M与附录的拼接：$M||((R,S),text)$。

6.3.5 验证过程

6.3.5.1 概述

在验证已签名消息的签名之前，验证方需要拥有 p、q 和 G 的可信副本。验证方还需获取验证过程所需的数据项：例如，验证密钥 Y（参见 ISO/IEC 14888-1:2008 第 9 章，了解所需的附加数据项）。

6.3.5.2 恢复见证值

验证方从附录中检索见证值 R 和签名第二部分 S。然后验证方检查以下条件是否成立：

——$0<S<q$；

——如果值γ的长度不大于β，则 R 的比特长度等于所用杂凑函数 h 的输出比特长度；

——如果值γ的长度大于β，则 R 的比特长度等于β。如果以上任一条件不成立，则应拒绝该签名。

6.3.5.3 准备待验证消息

验证方从已签名消息中检索$M_2=M$。M_1为空。

6.3.5.4 恢复赋值

本阶段与6.3.4.5相同。赋值函数的输入由6.3.5.2中的见证值R和6.3.5.3中的M_2组成。赋值$T=(T_1,T_2)$作为赋值函数（6.3.4.5）的输出重新计算。

6.3.5.5 重新计算预签名

此阶段的输入为：域参数、验证密钥Y、6.3.5.4中的赋值$T=(T_1,T_2)$以及6.3.5.2中的签名第二部分S。验证方利用公式(26)重新计算预签名的值Π'：

$$\Pi'=Y^{S\ mod\ q}G^{T_1\ mod\ q}\ mod\ p \tag{26}$$

6.3.5.6 重新计算见证值

此阶段的计算与 6.3.4.4 相同。验证方执行见证函数。输入为 6.3.5.5 中的Π'。注意M_1为空。输出为重新计算的见证值R'。

6.3.5.7 验证见证值

验证方将来自 6.3.5.6 的重新计算的见证值R'与来自 6.3.5.2 的R值进行比较。若$R' = R$，则签名有效。

6.4 Pointcheval/Vaudenay 算法

6.4.1 概述

Pointcheval/Vaudenay 方法是一种 DSA 算法的变体，其中$E = Z_p^*$，p 为素数，q为$p-1$的一个素因子。参数D等于 1。消息准备要求M_1为空，M_2为待签名消息，即$M_2 = M$。见证值由公式(27)定义：

$$R = \Pi \bmod q \tag{27}$$

赋值函数由公式(28)定义：

$$(T_1, T_2) = (-R, -H) \tag{28}$$

其中$H = h(I2BS(\beta, R)||M)$是见证值R与消息M拼接后的杂凑值。注意，上述T_2的计算需要将杂凑值转换为整数。转换函数见附录 B。

Pointcheval/Vaudenay 签名公式的系数(A, B, C)设置为公式(29)：

$$(A, B, C) = (S, T_1, T_2) \tag{29}$$

因此，签名公式变为公式 (30)：

$$SK - RX - H \equiv 0 (\bmod q) \tag{30}$$

注：本机制基于 D.Pointcheval 和 S.Vaudenay 设计的算法，见参考文献[31]

6.4.2 参数

p 素数

q p–1 的一个素因子

F 满足 $1 < F < p-1$ 和$F^{(p-1)/q} \bmod p > 1$ 的整数

G $F^{(p-1)/q} \bmod p$。

杂凑函数标识符或指定杂凑函数的 OID。

建议所有用户检查公共参数的正确生成。

6.4.3 签名密钥和验证密钥的生成

签名实体的签名密钥是一个秘密生成的随机或伪随机整数X，满足$0 < X < q$。参数D为 1。相应的公开验证密钥Y由公式(31)给出：

$$Y = G^X \bmod p \tag{31}$$

用户的签名密钥X和验证密钥Y通常在一段时间内固定。签名密钥X应保密。

6.4.4 签名过程

6.4.4.1 生成随机化值

签名实体计算一个随机或伪随机整数K，使得$0 < K < q$。

6.4.4.2 生成预签名

此阶段的输入是随机化值K，签名实体计算公式(32)：

$$\Pi = G^{K} \bmod p \tag{32}$$

6.4.4.3 准备待签名消息

对消息进行准备，使得M_1为空，M_2为待签名的消息，即$M_2 = M$。

6.4.4.4 计算见证值

签名实体计算$R = \Pi \bmod q$，其中见证值仅为预签名的函数。因此，参见公式(33)：

$$R = (G^{K} \bmod p) \bmod q \tag{33}$$

6.4.4.5 计算赋值

签名实体计算赋值$(T_1, T_2) = (-R, -BS2I(\gamma, H))$，其中 $H=h(\mathrm{I2BS}(\beta,R)||M_2)$为见证值与消息$M_2$拼接的杂凑码。在拼接前，见证值应转换为长度为$|p|$的比特串。

6.4.4.6 计算签名值

签名为(R, S)，其中 R 按 6.4.4.4 计算，且见公式(34)：

$$S = K^{-1}(BS2I(\gamma, H) + XR) \bmod q \tag{34}$$

6.4.4.7 构造附录

附录将由(R, S)与可选文本字段$text$的连接$((R, S), text)$组成。

6.4.4.8 构造已签名消息

已签名消息是消息M与附录的拼接：$M||((R, S), text)$。

6.4.5 验证过程

6.4.5.1 概述

在验证已签名消息的签名之前，验证方需要拥有p、q和G的可信副本。验证方还需获取验证过程所需的数据项：例如，验证密钥Y（其他所需数据项见 ISO/IEC 14888-1:2008 第 9 章）。

6.4.5.2 恢复见证值

验证方从附录中检索见证值R和签名的第二部分S。然后验证方检查是否满足 $0 < R < q$ 和 $0 < S < q$。若有任一条件不满足，则应拒绝该签名。

6.4.5.3 准备待验证消息

验证方从已签名消息中检索$M_2 = M$。M_1为空。

6.4.5.4 恢复赋值

此阶段与 6.4.4.5 相同。赋值函数的输入包括来自 6.4.5.2 的见证值R和来自 6.4.5.3 的M_2。赋值$T = (T_1, T_2)$作为赋值函数的输出被重新计算，见 6.4.4.5。

6.4.5.5 重新计算预签名

此阶段的输入为域参数、验证密钥Y、来自 6.4.5.4 的赋值$T = (T_1, T_2)$以及来自 6.4.5.2 的签名的第二部分S。验证方通过使用公式(35)计算预签名的一个重新计算值：

$$\Pi' = Y^{-S^{-1}T_1 \bmod q} G^{-S^{-1}T_2 \bmod q} \bmod p \quad (35)$$

6.4.5.6 重新计算见证值

此阶段的计算与 6.4.4.4 相同。验证方执行见证函数。输入为来自 6.4.5.5 的Π'。注意M_1为空。输出为重新计算的见证值R'。

6.4.5.7 验证见证值

验证方将来自 6.4.5.6 的重新计算的见证值R'与来自 6.4.5.2 的 R 值进行比较。若$R' = R$，则签名有效。

6.5 SDSA

6.5.1 概述

SDSA（Schnorr 数字签名算法）是一种签名机制，其中$E = Zp*$，p 为素数，q为整除$p-1$的素数。参数D等于 1。消息准备方式为：M_1为待签名消息，即$M_1 = M$，M_2为空。见证函数通过将R设为杂凑码来定义。赋值函数通过将$T_1 = -1$，并将T_2设为根据附录 B 给出的转换规则 BS2I 将R转换为整数后模q所得整数的负数来定义。SDSA 签名公式的系数(A, B, C)按公式(36)设置：

$$(A, B, C) = (T_1, T_2, S) \quad (36)$$

因此，签名公式变为公式(37)：

$$-K + T_2 X + S \equiv 0 (\bmod q) \quad (37)$$

注：SDSA 代表 Schnorr 数字签名算法。本机制取自参考文献[32]（许可自）和[33]。为与 ISO/IEC 14888 系列所用符号保持一致，此处对参考文献[32]和[33]中的符号作了调整。

6.5.2 参数

p　　一个素数，$2^{\alpha-1} < p < 2^{\alpha}$。

q　　$p-1$ 的一个质因数，$2^{\beta-1} < q < 2^{\beta}$。

G　　q 阶子群的生成元，满足 $1 < G < q$。

SDSA 允许四种(α, h)组合，即$(1\,024, \mathrm{SHA}-1)$、$(2\,048, \mathrm{SHA}-224)$、$(2\,048, \mathrm{SHA}-256)$和$(3\,072, \mathrm{SHA}-256)$。相应的$\beta$应根据表 1 中的$\alpha$选取。整数$p$、$q$和$G$可公开，并可为用户组共用。参数$p$、$q$和$G$按附录 D 的规定生成。参数$p$和$q$可使用 ISO/IEC 18032 中给出的素数生成技术生成。建议所有用户按照参考文献[17]检查 SDSA 公开参数是否正确生成。

建议基于 SHA-1 的数字签名仅用于遗留应用。

6.5.3 签名密钥和验证密钥的生成

签名实体的签名密钥为秘密生成的随机或伪随机整数X，满足 $0 < X < q$。参数D为 1。对应的验证密钥Y为公式(38)：

$$Y = G^X \bmod p \quad (38)$$

用户的签名密钥X和验证密钥Y通常在一段时间内固定。签名密钥X应保密。

6.5.4 签名过程

6.5.4.1 生成随机化值

签名实体计算一个随机或伪随机整数K，使得 $0 < K < q$。

6.5.4.2 生成预签名

此阶段的输入为随机化值K，签名实体计算公式(39)：

$$\Pi = G^{K} \bmod p \tag{39}$$

6.5.4.3 准备签名消息

消息准备使得M_1为待签名的消息，即$M_1 = M$，且M_2为空。

6.5.4.4 计算见证值

签名实体将见证值 R 计算为预签名Π和消息M_1的杂凑码[见公式(40)]：

$$R = h(\mathrm{I2BS}(\alpha, \Pi) \,\|\|\, M) \tag{40}$$

6.5.4.5 计算赋值

根据附录 B 中的转换规则 BS2I 将见证值R的值转换为整数，然后模q约简。赋值(T_1, T_2)为$(-1, -\mathrm{BS2I}(\gamma, R)\ mod\ q)$。

6.5.4.6 计算签名的第二部分

签名为(R, S)，其中$S = (K + BS2I(\gamma, R)X) \bmod q$。

可选地，可检查$R = 0$ 或$S = 0$ 是否成立。若任一条件成立，宜生成新的K值并重新计算签名（若签名正确生成，出现$R = 0$ 或$S = 0$ 的可能性极低）。

6.5.4.7 构造附录

附录将是(R, S)与可选文本字段$text$的拼接。

6.5.4.8 构造已签名消息

已签名消息是消息M与附录$M||((R, S)||text)$的拼接。

6.5.5 验证过程

6.5.5.1 概述

验证实体获取验证过程所需的数据项。

6.5.5.2 恢复见证值

验证方从附录中提取见证值R和签名的第二部分S。验证方检查R是否为杂凑函数范围内的非零字符串，且 $0 < S < q$。

6.5.5.3 准备待验证消息

验证方从已签名消息中获取M，并将消息分为两部分M_1和M_2，满足$M_1 = M$且M_2为空。

6.5.5.4 恢复赋值

赋值函数的输入包括来自 6.5.5.2 的见证值R。赋值为公式(41)：

$$T = (T_1, T_2) = (-1, -BS2I(\gamma, R) \bmod q) \tag{41}$$

6.5.5.5 重新计算预签名

此阶段的输入包括域参数、验证密钥Y、来自 6.5.5.4 的赋值 T=(T_1,T_2)以及来自 6.5.5.2 的签名第二部分S。验证方通过公式(42)计算预签名Π的重计算值

$$\Pi' = Y^{(T_2\ mod\ q)} G^{(-ST_1\ mod\ q)}\ mod\ p = Y^{(T_2\ mod\ q)} G^{(S\ mod\ q)}\ mod\ p \tag{42}$$

6.5.5.6 重新计算见证值

此阶段的计算与6.5.4.4和6.5.4.5相同。验证方执行见证函数。输入为来自6.5.5.5的Π'和来自6.5.5.3的M_1。输出为重计算的见证值R'，即重计算的预签名Π'和消息M的杂凑码。

6.5.5.7 验证见证值

验证方比较来自6.5.5.6的重计算见证值R'与来自6.5.5.2的已提取版R。如果$R' = R$,则签名验证通过。

6.6 EC-DSA

6.6.1 概述

EC-DSA（椭圆曲线数字签名算法）是DSA算法的椭圆曲线对应形式。EC-DSA签名公式的系数(A, B, C)按公式（43）设置：

$$(A, B, C) = (S, T_1, T_2) \tag{43}$$

其中$(T_1, T_2) = \big(-R, -\mathrm{BS2I}(\gamma, H)\big)$，$H = h(M)$是消息$M$的截断杂凑码，并按照附录B给出的转换规则转换为整数。杂凑函数h为ISO/IEC 10118-3中规定的SHA-1、SHA-224、SHA-256、SHA-384和SHA-512之一。

验证密钥Y为$Y = [X]G$，即参数D等于1。消息按如下方式准备：M_1为空，M_2为待签名消息，即$M_2 = M$。见证函数由公式（44）定义：

$$R = \mathrm{FE2I}(r, \Pi_X) \bmod q \tag{44}$$

转换规则FE2I见附录B。因此，签名公式变为公式（45）：

$$SK - RX - \mathrm{BS2I}(\gamma, H) \equiv 0 \pmod q \tag{45}$$

注：本机制基于参考文献[7]所述算法。

6.6.2 参数

F 有限域

E 有限域F上的椭圆曲线群

$\#E$ E的基数

q $\#E$的一个素因子

G 椭圆曲线上阶为q的点

上述所有参数均可公开，并可由一组用户共用。所用杂凑函数的安全强度应达到或高于与q的比特长度相对应的安全强度。

建议所有用户按照参考文献[7]或参考文献[17]检查EC-DSA公开参数是否正确生成。

建议仅在遗留应用中使用基于SHA-1的数字签名。

6.6.3 签名密钥和验证密钥的生成

签名者的签名密钥X是秘密生成的随机或伪随机整数，满足$0 < X < q$。参数D为1。对应的验证密钥Y由公式（46）给出：

$$Y = [X]G \tag{46}$$

用户的签名密钥X和验证密钥Y通常在一段时间内保持不变。签名密钥X应保密。

6.6.4 签名过程

6.6.4.1 生成随机数

签名者生成随机或伪随机整数K，满足$0 < K < q$。

6.6.4.2 生成预签名

本阶段的输入为随机数K，签名者按公式（47）计算：

$$\Pi = [K]G \tag{47}$$

6.6.4.3 准备待签名消息

对消息进行准备，使M_1为空，M_2为待签名消息，即$M_2 = M$。

6.6.4.4 计算见证值

签名者计算$R = \mathrm{FE2I}(r, \Pi_X) \bmod q$。

6.6.4.5 计算赋值

签名者计算杂凑码。若所选杂凑函数的输出比特长度大于$\lceil \log_2 q \rceil$，则将H置为$h(M_2)$最左侧（最高有效）的$\lceil \log_2 q \rceil$个比特；否则，H置为$h(M_2)$。随后，按照附录 B 中的转换规则 BS2I 将H转换为整数。赋值$(T_1, T_2) = \big(-R, -\mathrm{BS2I}(\gamma, H)\big)$。

6.6.4.6 计算签名的第二部分

签名为(R, S)，其中R按6.6.4.4计算，S按公式（48）计算：

$$S = K^{-1}\big(XR + \mathrm{BS2I}(\gamma, H)\big) \bmod q \tag{48}$$

应检查$R = 0$或$S = 0$是否成立。若$R = 0$或$S = 0$，宜生成新的K值并重新计算签名（若签名正确生成，则出现$R = 0$或$S = 0$的概率极低）。

6.6.4.7 构造附录

附录由(R, S)与可选文本字段$text$连接而成，即$\big((R, S), text\big)$。

6.6.4.8 构造已签名消息

已签消息由消息M与附录连接而成：

$$M \parallel \big((R, S), text\big)$$

6.6.5 验证过程

6.6.5.1 概述

验证方获取验证过程所需的数据元。

6.6.5.2 提取见证值

验证方从附录中提取见证值R和签名的第二部分S。验证方首先检查$0 < R < q$和$0 < S < q$；若任一条件不满足，则应拒绝该签名。

6.6.5.3 准备待验证消息

验证方从已签消息中检索M，并将消息分为两部分M_1和M_2。M_1为空，且$M_2 = M$。

6.6.5.4 检索赋值

本阶段与 6.6.4.5 相同。赋值函数的输入为来自 6.6.5.2 的见证值R和来自 6.6.5.3 的M_2。按照 6.6.4.5 的赋值函数重新计算赋值$T = (T_1, T_2)$。

6.6.5.5 重新计算预签名

本阶段的输入为系统参数、验证密钥Y、来自6.6.5.4的赋值$T=(T_1,T_2)$以及来自6.6.5.2的签名第二部分S。验证方按公式（49）重新计算预签名，得到Π'：

$$\Pi'=\left[-S^{-1}T_1 \bmod q\right]Y+\left[-S^{-1}T_2 \bmod q\right]G \quad (49)$$

6.6.5.6 重计算见证值

本阶段与6.6.4.4相同。验证方运行见证函数。输入为来自6.6.5.5的Π'，输出为重新计算的证据R'。

6.6.5.7 验证见证值

验证方将6.6.5.6的重计算的见证值R'与来自6.6.5.2的证据R进行比较。若$R'=R$，则签名验证通过。

6.7 EC-KCDSA

6.7.1 概述

EC-KCDSA（椭圆曲线韩国基于证书的数字签名算法）是一种验证密钥为$Y=[X^{-1}]G$的签名机制，即参数$D=-1$。消息按如下方式准备：M_1为空，M_2为待签名消息，即$M_2=M$。见证函数由公式（50）定义：

$$R=h\left(\mathrm{FE2BS}(r,\Pi_X)\right) \quad (50)$$

若γ大于β，则见证函数替换为公式（51）：

$$R=\mathrm{I2BS}\quad\left(\beta',\mathrm{BS2I}\quad\left(\gamma,h\left(\mathrm{FE2BS}(r,\Pi_X)\right)\right)\bmod 2^{\beta'}\right) \quad (51)$$

其中$\beta'=8*\lceil\beta/8\rceil$。

域参数应指明所用的杂凑函数。赋值函数由公式（52）定义：

$$(T_1,T_2)=(V,-1) \quad (52)$$

其中$V=\mathrm{BS2I}(\beta',R\oplus H) \bmod q$。$H$是由公钥$Y$和消息$M$计算得到的杂凑码。EC-KCDSA签名公式的系数$(A,B,C)$按公式（53）设置：

$$(A,B,C)=(T_2,S,T_1) \quad (53)$$

因此，签名公式变为公式（54）：

$$-K+SX^{-1}+V\equiv 0\ (\mathrm{mod}\ q) \quad (54)$$

注：本机制取自参考文献[37]。为与本文件其他部分所用符号保持一致，此处的符号与参考文献[37]略有不同。

6.7.2 参数

l 所选杂凑函数h的输入块长度（以比特为单位

F 有限域

E 有限域F上的椭圆曲线群

$\#E$ E的基数

q $\#E$的一个素因子

G 椭圆曲线上阶为q的点

杂凑函数标识符或指定杂凑函数的OID

上述所有参数均可公开，并可由一组用户共用。

建议所有用户按照参考文献[37]检查EC-KCDSA公开参数是否正确生成。

6.7.3 签名密钥和验证密钥的生成

签名实体的签名密钥是一个秘密生成的随机或伪随机整数X，满足$0 < X < q$。参数$D = -1$。对应的验证密钥Y由公式（55）给出：

$$Y = [X^{-1}]G \tag{55}$$

用户的签名密钥和验证密钥通常在一段时间内保持不变。签名密钥应保密。

6.7.4 签名过程

6.7.4.1 生成随机化值

签名实体计算一个随机或伪随机整K，使得 $0 < K < q$。

6.7.4.2 生成预签名

本阶段的输入为随机数K，签名者按公式（56）计算：

$$\Pi = [K]G \tag{56}$$

6.7.4.3 准备待签名消息

对消息进行准备，使M_1为空，M_2为待签名消息，即$M_2 = M$。

6.7.4.4 计算见证值

签名实体计算$R = h\big(\mathrm{FE2BS}(r, \Pi_X)\big)$，其中杂凑函数$h$的输出为$\Pi_X$的杂凑码。若$\gamma$大于$\beta$，则见证值的计算替换为$R = \mathrm{I2BS}\ \ \Big(\beta', \mathrm{BS2I}\ \ \big(\gamma, h\big(\mathrm{FE2BS}(r, \Pi_X)\big)\big)\bmod 2^{\beta'}\Big)$。

转换规则 FE2BS、I2BS 和 BS2I 见附录 B。

6.7.4.5 计算赋值

签名实体计算赋值$(T_1, T_2) = (V, -1)$，其中$V = \mathrm{BS2I}(\beta', R \oplus H) \bmod q$，且$H = h(Y' \| M_2)$为$Y'$与消息$M_2$连接所得比特串的杂凑码。

Y'为比特序列$\mathrm{FE2BS}(r, Y_X) \| \mathrm{FE2BS}(r, Y_Y)$最左侧的$l$个比特。若该比特序列的长度小于$l$，则在序列末尾按需补0至$l$比特。计算$V$时，应先将比特串$R \oplus H$转换为整数，再对$q$取模。

若γ大于β，则H的计算替换为$H = \mathrm{I2BS}\ \ \big(\beta', \mathrm{BS2I}\ \ \big(\gamma, h(Y' \| M_2)\big) \bmod 2^{\beta'}\big)$。

注：Y'对于某一用户是固定值，因此可将该值作为用户参数保存。

6.7.4.6 计算签名的第二部分

签名为(R, S)，其中R按6.7.4.4计算，S按公式（57）计算：

$$S = X(K - V) \bmod q \tag{57}$$

6.7.4.7 构造附录

附录由(R, S)与可选文本字段$text$连接而成，即$\big((R, S), text\big)$。

6.7.4.8 构造已签名消息

已签消息由消息M与附录连接而成：

$$M \| \big((R, S), text\big)$$

6.7.5 验证过程

6.7.5.1 概述

在验证已签消息的签名之前，验证方应具有E、q和G的可信副本。

验证方还应获取验证过程所需的数据元，例如验证密钥Y（其他所需数据元见 ISO/IEC 14888-1:2008 第 9 章）。

6.7.5.2 恢复见证值

验证方从附录中获取见证值R和签名的第二部分S。验证方检查下列条件是否成立：

——$0 < S < q$；

——若γ不大于β，则R的比特长度等于所用杂凑函数h的输出比特长度；

——若γ大于β，则R的比特长度等于β'。

若上述任一条件不成立，则应拒绝该签名。

6.7.5.3 准备待验证消息

验证方从已签消息中检索$M_2 = M$。M_1为空。

6.7.5.4 提取赋值

本阶段与 6.7.4.5 相同。赋值函数的输入由 6.7.5.2 的见证值R和来自 6.7.5.3 的M_2组成。按照 6.7.4.5 的赋值函数重新计算赋值$T = (T_1, T_2)$。

6.7.5.5 重新计算预签名

本阶段的输入为域参数、验证密钥Y、来自6.7.5.4的赋值$T = (T_1, T_2)$以及来自6.7.5.2的签名第二部分S。验证方按公式（58）重新计算预签名，得到Π'：

$$\Pi' = [S \bmod q]Y + [T_1 \bmod q]G \qquad (58)$$

6.7.6 重新计算见证值

本阶段的计算与 6.7.4.4 相同。验证方执行见证函数。输入为来自 6.7.5.5 的Π'。注意，M_1为空。输出为重新计算的见证值R'。

6.7.6.1 验证见证值

验证方将来自 6.7.5.6 的重新计算见证值R'与来自 6.7.5.2 的见证值R进行比较。若$R' = R$，则签名有效。

6.8 EC-GDSA

6.8.1 概述

EC-GDSA（椭圆曲线德国数字签名算法）是一种验证密钥为$Y = [X^{-1}]G$的签名机制，即参数D等于-1。消息准备如下：M_1为空，M_2为待签名消息，即$M_2 = M$。见证函数由公式(59)定义：

$$R = FE2I(r, \Pi_X) mod\ q \qquad (59)$$

EC-GDSA 签名公式的系数(A, B, C)设置如公式(60)所示：

$$(A, B, C) = (T_1, S, T_2) \qquad (60)$$

其中

$(T_1, T_2) = (-R, H)$;

H是消息M的杂凑码。因此，签名公式变为公式(61)：

$$-RK + SX^{-1} + H \equiv 0 \bmod q \qquad (61)$$

注：EC-GDSA 代表椭圆曲线德国数字签名算法[16]。

6.8.2 参数

F 有限域

E 域F上的椭圆曲线群

$\#E$ E的基数

q $\#E$的一个素因子

G 椭圆曲线上阶为q的点

杂凑函数标识符或带指定杂凑函数的 OID

所有这些参数均可公开，且可以为一组用户所共有。

建议所有用户检查公共参数生成的正确性。

6.8.3 签名密钥和验证密钥的生成

签名实体的签名密钥是秘密生成的随机或伪随机整数X,满足$0 < X < q$。参数D等于-1。因此，对应的验证密钥 Y 如公式(62)所示：

$$Y = [X^{-1}]G \tag{62}$$

用户的签名密钥X和验证密钥Y通常在一段时间内固定。签名密钥X应保密。

6.8.4 签名过程

6.8.4.1 生成随机化值

签名实体计算一个随机或伪随机整数K，使得 $0 < K < q$。

6.8.4.2 生成预签名

此阶段的输入是随机化值K，签名实体按公式(63)计算：

$$\Pi = [K]G \tag{63}$$

6.8.4.3 准备待签名消息

对消息进行准备，使得M_1为空，M_2为待签名的消息，即$M_2 = M$。

6.8.4.4 计算见证值

签名实体计算$R = FE2I\left(r, \Pi_X\right) mod\ q$，其中见证值为预签名的简单函数。

6.8.4.5 计算赋值

签名实体计算赋值$(T_1, T_2) = \left(-R, BS2I(\gamma, H)\right)$，其中$H$是消息$M_2$的杂凑值。

6.8.4.6 计算签名的第二部分

签名为(R, S)，其中 R 按 6.8.4.4 计算，且[见公式(64)]：

$$S = X(KR - BS2I(\gamma, H)) \bmod q \tag{64}$$

6.8.4.7 构造附录

附录将由(R, S)与可选文本字段$text$的连接组成，即$\left((R, S), text\right)$。

6.8.4.8 构造已签名消息

已签名消息是消息M与附录的连接：$M||\left((R, S), text\right)$。

6.8.5 验证过程

6.8.5.1 通用要求

验证实体获取验证过程所需的数据项。

6.8.5.2 恢复见证值

验证方从附录中提取见证值R和签名的第二部分S。验证方首先检查 $0 < R < q$和 $0 < S < q$；若任一条件检查失败，则应拒绝该签名。

6.8.5.3 准备待验证消息

验证方从已签名消息中提取M，并将消息分为两部分M_1和M_2。M_1为空且$M_2 = M$。

6.8.5.4 恢复赋值

此阶段与 6.8.4.5 相同。赋值函数的输入包括来自 6.8.5.2 的见证值R和来自 6.8.5.3 的M_2。赋值 T=(T_1, T_2)作为赋值函数 6.8.4.5 的输出被重新计算。

6.8.5.5 重计算预签名

此阶段的输入是系统参数、验证密钥Y、来自 6.8.5.4 的赋值$T = (T_1, T_2)$以及来自 6.8.5.2 的签名第二部分S。验证者通过是用公式（65）计算获得预签名的重新计算值Π'

$$\Pi' = [(-T_1)^{-1} S \bmod q]Y + [(-T_1)^{-1} T_2 \bmod q]G \tag{65}$$

6.8.5.6 重计算见证值

此阶段的计算与 6.8.4.4 相同。验证方执行见证函数。输入为 6.8.5.5 中的Π'。输出为重新计算的见证值R'。

6.8.5.7 验证见证值

验证方将 6.8.5.6 中重新计算的见证值R'与 6.8.5.2 中提取到的R进行比较。若$R' = R$，则签名验证通过。

6.9 EC-RDSA

6.9.1 通用要求

EC-RDSA（椭圆曲线俄罗斯数字签名算法）是一种签名机制，其验证密钥$Y = [X]G$；即参数D等于 1。消息准备如下：M_1为空，M_2为待签名消息，即$M_2 = M$。EC-RDSA 签名公式的系数(A, B, C)按公式(66)设置：

$$(A, B, C) = (T_1, T_2, -S) \tag{66}$$

其中$(T_1, T_2) = (H, R)$，且$H = h(M)$为消息M的杂凑值，按 6.9.4.5 所述转换为整数。见证函数由公式(67)定义：

$$R = FE2I(r, \Pi_X) \bmod q \tag{67}$$

因此签名公式变为公式(68)：

$$HK + RX - S \equiv 0 \bmod q \tag{68}$$

注：EC-RDSA 代表椭圆曲线俄罗斯数字签名算法。该机制取自参考文献[21]。此处符号已根据参考文献[22]进行了更改，以符合 ISO/IEC 14888（所有部分）中使用的符号。

6.9.2 参数

p *素数*

E 域$GF(p)$上的椭圆曲线群

$\#E$ E的基数

q $\#E$的一个素因子

G 椭圆曲线上阶为q的点

杂凑函数标识符或指定杂凑函数的 OID。

所有这些参数均可公开，且可以为一组用户所共用。

6.9.3 签名密钥和验证密钥的生成

签名实体的签名密钥是秘密生成的随机或伪随机整数X，满足 $0 < X < q$。参数D等于 1。对应的验证密钥Y为公式(69)：

$$Y = [X]G \tag{69}$$

用户的签名密钥X和验证密钥Y通常在一段时间内固定。签名密钥X应保密。

注：参考文献[21]未完整规定用户签名密钥X的生成过程。

6.9.4 签名过程

6.9.4.1 生成随机化值

签名实体生成一个随机或伪随机整数K，满足 $0 < K < q$。

6.9.4.2 生成预签名

此阶段的输入是随机化值K，签名实体计算公式(70)：

$$\Pi = [K]G \tag{70}$$

6.9.4.3 准备待签名消息

对消息进行准备，使得M_1为空，M_2为待签名的消息，即$M_2 = M$。

6.9.4.4 计算见证值

签名实体计算$R = FE2I(r, \Pi X) \bmod q$。

6.9.4.5 计算赋值

签名实体计算$H = h(M_2)$。然后根据附录 B 中的转换规则 BS2I 将H转换为整数。若$H \bmod q$等于 0，则赋值H为 1，若$BS2I(\gamma, H) \neq 0 \bmod q$，则赋值$(T_1, T_2)$为$(\mathrm{BS2I}(\gamma, H), R)$，否则赋值为$(1, R)$。

6.9.4.6 计算签名的第二部分

签名为(R, S)，其中 R 按 6.9.4.4 计算，且[见公式(71)]：

$$S = RX + KH \bmod q \tag{71}$$

签名者宜检查$R = 0$ 或$S = 0$。若$R = 0$ 或$S = 0$，宜生成新的K值并重新计算签名。

6.9.4.7 构造附录

附录将是(R, S)与可选文本字段$text$的拼接，即等于$((R, S)||text)$。

6.9.4.8 构造已签名消息

已签名消息是消息M与附录的拼接：$M||((R,S)||text)$。

6.9.5 验证过程

6.9.5.1 恢复见证值

验证方从附录中提取见证值R和签名的第二部分S。然后验证方检查 $0 < R < q$且 $0 < S < q$；若任一条件检查失败，则应拒绝该签名。

6.9.5.2 准备待验证消息

验证方从已签名消息中提取M，并将消息分为两部分M_1和M_2。M_1为空且$M_2 = M$。

6.9.5.3 恢复赋值

本阶段与 6.9.4.5 相同。赋值函数的输入包括来自 6.9.5.1 的见证值R和来自 6.9.5.2 的M_2。赋值$T = (T_1, T_2)$按 6.9.4.5 中给出的赋值函数重新计算为输出。

6.9.5.4 重计算预签名

本阶段的输入包括系统参数、验证密钥Y、来自 6.9.5.3 的赋值$T = (T_1, T_2)$以及来自 6.9.5.1 的签名的第二部分S。验证方通过使用公式(72)计算预签名，得到重新计算的值Π'。

$$\Pi' = [-T_1^{-1}T_2 \bmod q]Y + [T_1^{-1}S \bmod q]G \quad (72)$$

6.9.5.5 重计算见证值

本阶段的计算与 6.9.4.4 相同。验证方执行见证值函数。输入是来自 6.9.5.4 的Π'。输出是重新计算的见证值R'。

6.9.5.6 验证见证值

验证方将 6.9.5.5 中重新计算的见证值R'与 6.9.5.1 中提取到的R进行比较。若$R' = R$，则签名验证通过。

6.10 EC-SDSA

6.10.1 通用要求

EC-SDSA（椭圆曲线 Schnorr 数字签名算法）是一种验证密钥为$Y = [X]G$的签名机制；即参数D等于 1。消息准备使得M_2为空，$M_1 = M$为待签名消息。见证值R通过两种方法之一计算为消息M和随机预签名$\Pi = [K]G$的杂凑码，即：

——标准方法：$R = h(FE2BS(r, \Pi_X)||\mathrm{FE2BS}(r, \Pi_Y)||M)$；或

——优化方法：$R =$——优化方法：$R = h(FE2BS(r, \Pi_X)||M)$。

第一种方法通过对Π的横坐标、Π的纵坐标以及消息M的连接结果进行杂凑计算来生成见证值。第二种方法在杂凑计算中省略纵坐标，从而提升性能。第二种方法是 EC-SDSA 的一种优化变体（参见参考文献[30]）。EC-SDSA 签名公式的系数(A, B, C)设置为公式(73)：

$$(A, B, C) = (T_1, T_2, S) \quad (73)$$

其中$(T_1, T_2) = (-1, -BS2I(\gamma, R) \bmod q)$。

因此，签名公式变为公式(74)：

$$-K + T2X + S \equiv 0(\bmod q) \tag{74}$$

注：EC-SDSA 代表椭圆曲线 Schnorr 数字签名算法。该机制取自参考文献[33]。此处符号已按参考文献[33]进行修改，以符合 ISO/IEC 14888 系列中使用的符号。

6.10.2 参数

F 为有限域
E 为域F上的椭圆曲线群
$\#E$ E的阶
q 是$\#E$的素因子
G 是椭圆曲线上的一个q阶点，
杂凑函数标识符或指定杂凑函数的 OID。
所有这些参数可以是公开的，并且可以为一组用户所共用。
建议所有用户根据参考文献[7]检查公开参数的正确生成。

6.10.3 签名密钥和验证密钥的生成

签名实体的签名密钥是一个秘密生成的随机或伪随机整数X，满足 $0 < X < q$。参数D为 1。对应的验证密钥Y为$Y = [X]G$。

用户的签名密钥X与验证密钥Y通常在一段时间内固定。签名密钥X应保密。

6.10.4 签名过程

6.10.4.1 生成随机化值

签名实体计算一个随机或伪随机整数K，使得 $0 < K < q$。

6.10.4.2 生成预签名

此阶段的输入是随机化值K，签名实体计算$\Pi = [K]G$。

6.10.4.3 准备待签名消息

准备消息，使得M_1为待签名的消息，即$M_1 = M$，且M_2为空。

6.10.4.4 计算见证值

签名实体计算$R = h(\mathrm{FE2BS}(r, \Pi_X)||\mathrm{FE2BS}(r, \Pi_Y)||M)$。

对于 EC-SDSA 的优化变体，签名实体计算$R = h(\mathrm{FE2BS}(r, \Pi_X)||M)$

6.10.4.5 计算赋值

见证值R根据附录 B 中的转换规则将 BS2I 转换为整数，然后模q归约。(T_1, T_2)赋值为$(-1, -\mathrm{BS2I}(\gamma, R) \bmod q)$。

6.10.4.6 计算签名的第二部分

签名值为(R, S)，其中$S = (K + \mathrm{BS2I}(\gamma, R)X) \bmod q$。

作为一个选项，可以检查$R = 0$ 或$S = 0$。若$R = 0$ 或$S = 0$，则应生成新的K值并重新计算签名（如果签名正确生成，$R = 0$ 或$S = 0$ 的可能性极小）。

6.10.4.7 构造附录

附录是(R,S)与可选文本字段$text$的拼接。

6.10.4.8 构造已签名消息

已签名消息是消息M与附录的拼接：$M||\big((R,S)||text\big)$。

6.10.5 验证过程

6.10.5.1 通用要求

验证实体获取验证过程所需的数据项。

6.10.5.2 恢复见证值

验证方从附录中获取见证值R和签名的第二部分S。验证方首先检查R是否为杂凑函数范围内的非零字符串且$0 < S < q$；若任一条件违反，则应拒绝该签名。

6.10.5.3 准备待验证消息

验证方从已签名消息中获取M，并将消息分为两部分M_1和M_2。$M_1 = M$且M_2为空。

6.10.5.4 恢复赋值

赋值函数的输入包括来自 6.10.5.2 的见证值R。赋值$T = (T_1, T_2) = (-1, -\text{BS2I}(\gamma, R) \bmod q)$。

6.10.5.5 重新计算预签名

此阶段的输入为系统参数、验证密钥Y、来自 6.10.5.4 的赋值$T = (T_1, T_2)$以及来自 6.10.5.2 的签名第二部分S。验证方通过使用公式计算得到预签名的重计算值Π'：

$$\Pi' = [-ST_1 \bmod q]G + [T_2 \bmod q]Y = [S \bmod q]G + [T_2 \bmod q]Y$$

6.10.5.6 重计算见证值

本阶段的计算与 6.10.4.4 和 6.10.4.5 相同。验证方执行来自 6.10.4.4 的见证函数。输入为来自 6.10.5.5 的Π'。输出为重计算的见证值R'，即重新计算的预签名Π'和消息M的杂凑码。

6.10.5.7 验证见证值

验证方将 6.10.5.6 中重新计算的见证值R'与检索到的R版本进行比较。如果$R' = R$，则签名验证通过。

6.11 EC-FSDSA

6.11.1 通用要求

EC-FSDSA（椭圆曲线全 Schnorr 数字签名算法）是一种签名机制，其验证密钥$Y = [X]G$，*即参数* D 等于 1。消息的制备方式为M_1为空，$M_2 = M$（待签名的消息）。见证值R按公式(75)计算：

$$R = \text{FE2BS}(r, \Pi_X)||FE2BS(r, \Pi_Y) \tag{75}$$

EC-FSDSA 签名公式的系数(A, B, C)按公式(76)设置：

$$(A, B, C) = (T_1, T_2, S) \tag{76}$$

其中$T=(T_1,T_2)=(-1,-BS2I(\gamma,h(R||M)) \bmod q)$。

因此，签名公式变为公式(77)：

$$-K+T_2X+S\equiv 0(\bmod q) \tag{77}$$

注：EC-FSDSA 代表椭圆曲线全 Schnorr 数字签名算法。该机制取自参考文献[33]。此处符号已从参考文献[33]更改，以符合 ISO/IEC 14888 系列中使用的符号。

6.11.2 参数

F 是一个有限域
E 是域F上的椭圆曲线群
$\#E$ E的基数
q $\#E$ 的一个素因子
G 椭圆曲线E上阶为q的点

杂凑函数标识符或指定杂凑函数的 OID；所有这些参数可以公开，并且可以是一组用户共有的。

建议所有用户根据参考文献[7]检查公开参数的正确生成。

6.11.3 签名密钥和验证密钥的生成

签名实体的签名密钥是一个秘密生成的随机或伪随机整数X，满足 $0<X<q$。参数D为 1。对应的验证密钥$Y=[X]G$。

用户的签名密钥X和验证密钥Y通常在一段时间内固定。签名密钥X应保密。

6.11.4 签名过程

6.11.4.1 生成随机化值

签名实体计算一个随机或伪随机整数K，使得 $0<K<q$。

6.11.4.2 生成预签名

本阶段的输入是随机化值K，签名实体计算$\Pi=[K]G$。

6.11.4.3 准备待签名消息

消息的准备方式为M_2是待签名的消息，即$M_2=M$，且M_1为空。

6.11.4.4 计算见证值

签名实体计算$R=\mathrm{FE2BS}(r,\Pi_X)||\mathrm{FE2BS}(r,\Pi_Y)$。

6.11.4.5 计算赋值

签名实体计算杂凑值$h(R||M)$。之后，按照附录 B 中的转换规则由 BS2I 将杂凑值转换为整数，再模q归约。赋值(T_1,T_2)为$(-1,-\mathrm{BS2I}(\gamma,h(R||M)) \bmod q)$。

6.11.4.6 计算签名的第二部分

签名值为(R,S)，其中$S=(K+BS2I(\gamma,h(R||M))X \bmod q)$。

可选地，可以检查$R=0$ 或$S=0$。如果$R=0$ 或$S=0$，则应生成新的K值并重新计算签

名（如果签名正确生成，则$R=0$或$S=0$的可能性极低）。

6.11.4.7 构造附录

附录是(R,S)与可选文本字段$text$的拼接。

6.11.4.8 构造已签名消息

已签名消息是消息M与附录的串联：$(M||(R,S)||text)$。

6.11.5 验证过程

6.11.5.1 通用要求

验证实体获取验证过程所需的数据项。

6.11.5.2 恢复见证值

验证方从附录中检索见证值R和签名第二部分S。验证方首先检查R表示E上的一个点，且$0<S<q$；若任一条件不满足，则应拒绝签名。

6.11.5.3 准备待验证消息

验证方从已签名消息中获取M，并将消息分为两部分M_1和M_2。$M_2=M$且M_1为空。

6.11.5.4 恢复赋值

赋值函数的输入根据 6.11.4.5 从 6.11.5.2 的见证值R和 6.11.5.3 的消息M计算得出。赋值由$T=(T_1,T_2)=\left(-1,-\mathrm{BS2I}\left(\gamma,h(R||M)\right)\ mod\ q\right)$给出。

6.11.5.5 重新计算预签名

该阶段的输入是系统参数、验证密钥Y、来自 6.11.5.4 的赋值$T=(T_1,T_2)$以及来自 6.11.5.2 的签名第二部分S。验证方通过使用以下公式计算得到预签名的重新计算值Π'：

$$\Pi'=[-ST_1 \bmod q]G+[T_2 \bmod q]Y=[S \bmod q]G+[T_2 \bmod q]Y$$

6.11.5.6 重新计算见证值

该阶段的计算与 6.11.4.4 相同。验证方执行见证函数。输入是来自 6.11.5.5 的Π'。输出是重新计算的见证值R'。

6.11.5.7 验证见证值

验证方将来自 6.11.5.6 的重新计算见证值R'与检索到的 R 版本进行比较，如果$R'=R$，则签名验证通过。

6.12 SM2

6.12.1 通用要求

SM2 算法是一种基于椭圆曲线的签名机制。在椭圆曲线上，参数D等于 1。消息准备使得M_1是杂凑值Z和待签名消息M的拼接，其中Z是消息的杂凑值，该消息由ID的长度（签名实体的标识符）、ID的值、a_1、a_2、G 的x坐标、G的y坐标、Y的x坐标和Y的y坐标拼接而成，即$M_1=Z||M$，且M_2为空。见证函数由以下公式定义：

$R=\mathrm{BS2I}(\gamma,h(M_1))+\mathrm{FE2I}(r,\Pi_X) \bmod q$

转换规则 BS2I 和 FE2I 在附录 B 中给出。赋值函数由公式(78)定义：

$$(T_1, T_2) = (-1, R) \tag{78}$$

SM2 签名公式的系数(A, B, C)按公式(79)设置：

$$(A, B, C) = (T_1, S + T_2, S) \tag{79}$$

因此，签名公式变为公式(80)：

$$-K + (R + S)X + S \equiv 0(\bmod q) \tag{80}$$

注：SM2 签名机制取自[42]。此处符号已从[42]更改，以符合本文件中使用的符号。

6.12.2 参数

F 一个有限域
E 域F上的椭圆曲线群
$\#E$ E的基数
q $\#E$的一个素因子
G 椭圆曲线上的一个q阶点
杂凑函数标识符或带有指定杂凑函数的 OID。
所有这些参数可以是公开的，并且可以是一组用户共用的。

注：建议所有用户检查公开参数的正确生成。

6.12.3 签名密钥和验证密钥的生成

签名实体的签名密钥是一个秘密生成的随机或伪随机整数 X，满足 $0 < X < q - 1$。参数D为 1。对应的验证密钥Y为公式(81)：

$$Y = [X]G \tag{81}$$

用户的签名密钥X和验证密钥Y通常在某个时间段内固定。签名密钥X应保密。

6.12.4 签名过程

6.12.4.1 生成随机化值

签名实体生成一个随机或伪随机整数K，满足 $0 < K < q$。

6.12.4.2 生成预签名

该阶段的输入是随机化值K，签名实体计算$\Pi = [K]G$。

6.12.4.3 准备待签名消息

令 *entlen* 为签名实体区分标识符ID的比特长度。令 *ENTL* 为从整数 *entlen* 转换得到的双字节字符串，即$ENTL = \mathrm{I2BS}(16, entlen)$。则$Z$可按公式(82)计算：

$$Z = h\big(ENTL||ID||\mathrm{FE2BS}(r, a_1)||\mathrm{FE2BS}(r, a_2)||\mathrm{FE2BS}(r, G_X)||\mathrm{FE2BS}(r, G_Y)||\mathrm{FE2BS}(r, Y_X)||\mathrm{FE2BS}(r, Y_Y)\big) \tag{82}$$

消息准备为M_1是杂凑码Z与待签名消息M的拼接，即$M_1 = Z||M$，*且*M_2为空。

转换规则 I2BS 和 FE2BS 见附录 B。

6.12.4.4 计算见证值

签名实体计算$H = h(M_1)$，*然后计算*$R = \left(\mathrm{BS2I}(\gamma, H) + \mathrm{FE2I}(r, \Pi_X)\right) mod\ q$。

6.12.4.5 计算赋值

签名实体计算赋值$(T_1, T_2) = (-1, R)$。

6.12.4.6 计算签名的第二部分

签名为(R, S)，其中按 6.12.4.4 计算R，并[见公式(83)]：

$$S = ((1 + X) - 1(K - RX)) \bmod q \tag{83}$$

需检查是否满足$R = 0$、$R + K = q$或$S = 0$。若$R = 0$、$R + K = q$或$S = 0$ 之一成立，则应生成新的K值并重新计算签名。

注 1：若正确生成签名，则$R = 0$、$R + K = q$或$S = 0$ 的可能性极低。

注 2：易知$R + S = (1 + X) - 1(R + K) \bmod q$。鉴于 $0 < R + K < 2q$，可得$R + S = 0(\bmod q)$且仅当$R + K = q$。

6.12.4.7 构造附录

附录由(R, S)与可选文本字段$text$的连接$\left((R, S), text\right)$组成。

6.12.4.8 构造已签名消息

已签名消息是消息M与附录的连接：$(M||(R, S), text)$。

6.12.5 验证过程

6.12.5.1 通用要求

验证实体获取验证过程所需的数据项。

6.12.5.2 恢复见证值

验证方从附录中检索见证值R和签名第二部分S。然后验证方首先检查是否满足 $0 < R < q$且 $0 < S < q$；若任一条件不满足，则应拒绝签名。

6.12.5.3 准备待验证消息

验证方从已签名消息中检索M，并将消息分为两部分M_1和M_2。

$M_1 = Z||M$，其中$Z = h\left(ENTL||ID||\mathrm{FE2BS}(r, a_1)||\mathrm{FE2BS}(r, a_2)||\mathrm{FE2BS}(r, G_X)||\mathrm{FE2BS}(r, G_Y)||\mathrm{FE2BS}(r, Y_X)||\mathrm{FE2BS}(r, Y_Y)\right)$，且$M_2$为空。

6.12.5.4 恢复赋值

赋值函数的输入包括来自 6.12.5.2 的见证值 R。赋值$T = (T_1, T_2) = (-1, R)$。

6.12.5.5 重新计算预签名

此阶段的输入为系统参数、验证密钥Y、6.12.5.4 的赋值$T=(T_1,T_2)$以及 6.12.5.2 的签名第二部分S。验证方计算$W=(T_2+\mathrm{S})\ mod\ q$，*并检查* W 是否等于 0。若$W=0$ 成立，则应拒绝该签名。验证方随后通过公式(84)重新计算得到预签名值Π'：

$$\Pi' = [\mathrm{S}]\mathrm{G} + [\mathrm{W}]\mathrm{Y} \tag{84}$$

6.12.5.6 重算见证值

本阶段的计算与 6.12.4.4 相同。验证方执行见证函数。输入为 6.12.5.5 的Π'和 6.12.5.3 的M_1。输出为重新计算的见证值R'。

6.12.5.7 验证见证值

验证方将来自 6.12.5.6 的重新计算的见证值R'与检索到的 R 版本进行比较。如果$R'=R$，则签名验证通过。

7 基于身份的机制

7.1 通用要求

第 7 章规定了三种基于双线性对的基于身份密码机制。基于身份签名机制涉及三个参与实体：可信密钥生成中心(KGC)、签名者和验证方。

签名过程需要以下数据项：

——域参数,$E, GF(r), G_1, G_2, q, P, <>$；

——公开主密钥V；

——签名密钥X；

——消息M；

——用于H_1和H_2的杂凑函数标识符（可选）；

——身份字符串ID；

——其他文本（可选）。

杂凑函数标识符可用于绑定签名机制与杂凑函数。

——验证过程需要以下数据项：域参数,$E, GF(r), G_1, G_2, q, P, <>$；

——公开主密钥V；

——验证密钥Y，可从身份字符串ID计算；

——消息M；

——签名 Σ；

——用于H_1和H_2的杂凑函数标识符（可选）；

注 1：签名者和验证方必须就机制中使用特定的杂凑函数h、H_1和H_2达成一致。如果其中任何杂凑函数未通过其他方式唯一确定，则在签名和验证过程中都需要杂凑函数标识符（参见 ISO/IEC 14888-1）。

——身份字符串ID；

——其他文本（可选）。

注 2：IBS-1 和 IBS-2 的典型椭圆曲线是$GF(r)$上的超奇异椭圆曲线，其中$r=p^m$，p为≥ 2的素数，m为≥ 1的整数。

7.2 IBS-1

7.2.1 通用要求

IBS-1 是基于椭圆曲线点加法群的基于身份的签名方案。它采用公式(85)：

$$(A,B,C)=(T_1,S,T_2) \tag{85}$$

其中$T_1=-Y,T_2=[R]Y.D=-1$。

因此，签名公式为公式(86)：

$$[-K]Y+[U^{-1}]S+[R]Y\equiv 0_E(inG_1) \tag{86}$$

注：本机制基于参考文献[22]中 Hess 所设计的算法。

7.2.2 参数

签名机制在涉及的实体共享以下参数的环境中进行，这些参数已在第 4 章中定义：$G_1,G_2,q,P,<>,H_1$和H_2。

建议所有用户检查公共参数的正确生成。

7.2.3 主密钥和签名/验证密钥的生成

KGC 的主密钥对为(U,V)，其中U是通过随机选择满足$0<U<q$的整数生成的主私钥，V是通过计算$V=[U]P$生成的主公钥。KGC 公布V并保密U。

签名者的签名和验证密钥对为(X,Y)，其中Y是从身份字符串ID和杂凑函数H_1生成的验证密钥，即$Y=H_1(ID)$，X是通过计算$X=[U]Y$生成的签名密钥，由 KGC 完成并交给签名者。

7.2.4 签名过程

7.2.4.1 生成随机化值

签名者首先选择一个随机或伪随机整数K，满足$0<K<q$。签名者将值K保密。

7.2.4.2 生成预签名

签名者将K、P和X作为输入，生成预签名结果[公式(87)]：

$$\Pi=<X,P>^K \tag{87}$$

注：Π是$GF(p^m)$的扩域中的元素，该扩域在特征$p=2$时次数为 4，在特征$p=3$时次数为 6，在特征$p>3$时次数为 2。

7.2.4.3 准备待签名消息

签名者准备待签名消息，使得M_2为空，M_1为已签名消息M，即$M_1=M$。

7.2.4.4 计算见证值

设$\Pi=(\Pi_a,\Pi_b)$，签名者对$M_1||FE2BS(r,\Pi_a)||FE2BS(r,\Pi_b)$（$M_1$，$FE2BS(r,\Pi_a)$和$FE2BS(r,\Pi_b)$的拼接）应用杂凑函数$H_2$，得到见证值[公式(88)]：

$$R=BS2I(\gamma,H_2(M_1||FE2BS(r,\Pi a)||FE2BS(r,\Pi b))\bmod q \tag{88}$$

如果$R=0$，输出“invalid”并停止。

对于更高扩展次数的域，待杂凑值中将出现更多项。例如，对于扩展次数3，$\Pi=(\Pi_a,\Pi_b,\Pi_c)$，H_2的输入为公式(89)：

$$M_1||FE2BS(r,\Pi_a)||FE2BS(r,\Pi_b)||FE2BS(r,\Pi_c) \tag{89}$$

7.2.4.5 计算赋值

赋值为$T=(T_1,T_2)$为$(-Y,[R]Y)$。然而，签名者无需计算该赋值。

7.2.4.6 计算签名的第二部分

签名者计算签名的第二部分见公式(90)：

$$S=[K-R]X \tag{90}$$

签名是$\Sigma=(R,S)$。

7.2.4.7 构造附录

签名者构造附录，其为(R,S)与可选文本字段$text$的拼接，即$\big((R,S),text\big)$。

7.2.4.8 构造已签名消息

已签名消息是消息M与附录的拼接，即$M||((R,S),text)$。

7.2.5 验证过程

7.2.5.1 通用要求

验证方首先获取验证过程所需的必要数据项。

7.2.5.2 恢复见证值

验证方从附录中提取见证值R和签名的第二部分S。然后验证方检查$S\in G1$是否成立；若条件不满足，则应拒绝该签名。否则，验证方执行以下步骤。

7.2.5.3 准备待验证消息

验证方从已签名消息中提取M，并将消息分为两部分M_1和M_2，使得M_2为空且M_1等于M。

7.2.5.4 恢复赋值

赋值 T=(T_1,T_2)，其中$T_1=-Y$，$T_2=[R]Y$。然而，验证方无需计算该赋值。

7.2.5.5 重计算预签名

验证方重新计算预签名值[见公式(91)]：

$$\Pi'=<S,P>*<Y,V>^R \tag{91}$$

7.2.5.6 重计算见证值

验证方重新计算见证值[见公式(92)]：

$$R'=BS2I(\gamma,H_2(M_1||FE2BS(r,\Pi_{a'})||FE2BS(r,\Pi_{b'})))\bmod q \tag{92}$$

对于更高扩展度的域，待杂凑值中将出现更多项。例如，对于扩展 3，$\Pi'=(\Pi_{a'},\Pi_{b'},\Pi_{c'})$，$H_2$的输入为公式(93)：

$$M_1||FE2BS(r,\Pi_{a'})||FE2BS(r,\Pi_{b'})||FE2BS(r,\Pi_{c'}) \tag{93}$$

7.2.5.7 验证见证值

验证方检查$R'=R$是否成立。若成立，则签名验证通过；否则无效。

7.3 IBS-2

7.3.1 概述

IBS-2 是一种基于椭圆曲线点加法群的基于身份的签名方案。它采用公式(94)：

$$(A,B,C)=(T_1,S,T_2) \tag{94}$$

其中

$T_1=-Y, T_2=[-H]Y$;

H是从H_2得到的杂凑码，$D=-1$。

因此，签名公式变为公式(95)：

$$[-K]Y+[U^{-1}]S+[-H]Y\equiv 0_E(in\ G_1) \tag{95}$$

注：本机制基于 Cha 和 Cheon 等人设计的算法。

7.3.2 参数

参数与 7.2.2 相同。

7.3.3 主密钥和签名/验证密钥的生成

该操作与 7.2.3 相同。

7.3.4 签名过程

7.3.4.1 生成随机化值

签名者首先选择一个随机或伪随机整数K，满足 $0<K<q$。签名者将值K保密。

7.3.4.2 生成预签名

签名者将K和Y作为输入，生成预签名结果[参见公式(96)]：

$$\Pi=[K]Y \tag{96}$$

7.3.4.3 准备待签名消息

签名者准备待签名消息，使得M_1为空，M_2为已签名消息M，即$M_2=M$。

7.3.4.4 计算见证值

签名者从预签名结果中获取见证值[参见公式(97)]：

$$R=\Pi \tag{97}$$

7.3.4.5 计算赋值

赋值为公式(98)：

$$T=(T_1,T_2) \tag{98}$$

其中：

$T_1=-Y$，$T_2=[-H]Y$。$H=BS2I(\gamma,H_2(M_2||FE2BS(r,\Pi_X)))\bmod q$。

然而，签名者仅需计算H的值。

7.3.4.6 计算签名的第二部分

签名者按照公式(99)计算签名的第二部分：

$$S = [K + H]X \tag{99}$$

签名是$\Sigma = (R, S)$。

7.3.4.7 构造附录

签名者构建附录，该附录是(R, S)与可选文本字段$text$的拼接，即$((R, S), text)$。

7.3.4.8 构造已签名消息

已签名消息是消息M与附录的拼接，即$M||((R, S), text)$。

7.3.5 验证过程

7.3.5.1 通用要求

验证方首先获取验证过程所需的必要数据项。

7.3.5.2 恢复见证值

验证方从附录中提取预签名R和签名的第二部分S。验证方首先检查R和$S \in G_1$是否成立；若任一条件不满足，则应拒绝签名。否则，验证方执行以下步骤。

7.3.5.3 准备待验证消息

验证方从已签名消息中提取M，并将消息分割为两部分M_1和M_2，使得M_1为空且M_2等于M。

7.3.5.4 恢复赋值

赋值如公式(100)所示：

$$T = (T_1, T_2) \tag{100}$$

其中：

$T_1 = -Y; T_2 = [-H]Y;\ H = BS2I(\gamma, H_2(M_2||FE2BS(r, \Pi X)))\ \mathrm{mod}\ q$。

然而，验证方仅需重新计算H的值。

7.3.5.5 重计算预签名

验证方按公式(101)获取预签名值：

$$\Pi' = R \tag{101}$$

7.3.5.6 重计算见证值

验证方不重新计算见证值R，而是计算两个配对$< P, S >$和$< V, \Pi' + [H]Y >$。

7.3.5.7 验证见证值

验证方检查$< P, S >=< V, \Pi' + [H]Y >$是否成立。若成立，则签名验证通过；否则，签名无效。

7.4 Chinese IBS

7.4.1 概述

中国IBS算法是一种基于身份的签名方案，基于椭圆曲线点的加法群。其采用公式(102)：

$$(A,B,C)=(T_1,S,[Y^{-1}]S+T_2) \quad (102)$$

其中：

$T_1=[-Y^{-1}]P, T_2=[Y^{-1}R]P, D=-1$。

因此，签名公式变为公式(103)：

$$[-KY^{-1}]P+[U^{-1}]S+[Y^{-1}]S+[Y^{-1}R]P\equiv 0_E(inG_1) \quad (103)$$

注：中国 IBS 签名机制取自参考文献[43]。为与本文件所用符号保持一致，此处对参考文献[43]中的符号作了调整。

7.4.2 参数

签名机制在如下环境中进行：所涉及的实体共享第 4 章中定义的以下参数：$G_1, G_2, P, \mathrm{Q}, q, <>$ 和h。

给定一个输出比特长度为v的杂凑函数h、一个比特长度为bn的非负整数n以及一个比特串Z，函数$h_i(Z,n)(i=1,2)$定义如下：

1) 设一个 32 比特构成的计数器$ct_1=0x00000001$，并且使$hlen=8\lceil(5*bn)/32\rceil$；其中，$\lceil z\rceil$ 表示不小于 z 的最小整数。

2) 从$j=1$到$\lceil hlen/v\rceil$执行：

2.1)计算$Ha_j=h(0x0i||Z||ct_j)$；

2.2)$ct_{j+1}=ct_j+1$;

3) 将Ha设为$Ha_1||Ha_2||\cdots||Ha_{\lceil hlen/v\rceil}$的前$hlen$比特。

4) 输出$BS2I(hlen,Ha)\bmod(n-1)+1$.

建议所有用户检查公开参数的正确生成。

7.4.3 主密钥和签名/验证密钥的生成

KGC 的主密钥对为(U,V)，其中U是通过随机选择满足 $0<U<q$的整数生成的主私钥，V是通过计算$V=[U]Q$生成的主公钥。KGC 发布主公钥V并主私钥U。

签名者的签名和验证密钥对为(X,Y)，其中Y是从身份字符串 *ID*、私钥生成函数标识符 *hid* 以及函数h_1生成的验证密钥，即$Y=h_1(ID||hid,q)$，X是通过计算$X=[U(U+Y)^{-1}P]$生成的签名密钥，由 KGC 完成并交给签名者。若$(U+Y)\bmod q=0$，则 KGC 生成另一个主密钥对，发布主公钥并更新私钥签名密钥。

7.4.4 签名过程

7.4.4.1 生成随机化值

签名者生成随机或伪随机整数K，满足 $0<K<q$。签名者保密K的值。

7.4.4.2 生成预签名

签名者将K、P和 V作为输入，生成预签名结果[见公式(104)]：

$$\Pi=<P,V>^K \quad (104)$$

注：$<P,V>$可以预先计算。

7.4.4.3 准备待签名消息

签名者准备消息，使M_2为空，M_1为已签名消息M，即$M_1 = M$。

7.4.4.4 计算见证值

令$\Pi = (\Pi_a, \Pi_b)$。签名者将杂凑函数h_2应用于M_1、$FE2BS(r, \Pi_a)$和$FE2BS(r, \Pi_b)$的拼接，以获得见证值[见公式(105)]：

$$R = h_2(M_1||FE2BS(r, \Pi a)||FE2BS(r, \Pi b), q) \quad (105)$$

如果$(R - K) \bmod q = 0$,应重新产生一个K的值，再次计算签名值。

对于更高扩张度的域，待杂凑值中将出现更多项。例如，对于扩张度 3，$\Pi = (\Pi_a, \Pi_b, \Pi_c)$，$h_2$的输入将为公式(106)：

$$M_1||FE2BS(r, \Pi a)||FE2BS(r, \Pi b)||FE2BS(r, \Pi c) \quad (106)$$

7.4.4.5 计算赋值

赋值为$T = (T_1, T_2)$为$([-Y^{-1}]P, [Y^{-1}R]P)$。但是，签名者不需要计算该赋值。

7.4.4.6 计算签名的第二部分

签名者计算签名的第二部分为公式(107)：

$$S = [R - K]X \quad (107)$$

签名为$\Sigma = (R, S)$.

7.4.4.7 构造附录

签名者构造附录，附录是(R, S)与可选文本字段$text$的拼接，即$((R, S), text)$。

7.4.4.8 构造已签名消息

已签名消息是消息M与附录的拼接，即$M||((R, S), text)$。

7.4.5 验证过程

7.4.5.1 通用要求

验证方首先获取验证过程所需的必要数据项。

7.4.5.2 恢复见证值

验证方从附录中提取见证值R和签名的第二部分S。然后验证方检查$R \in [1, q - 1]$和$S \in G_1$是否成立；如果任一条件不满足，则签名被拒绝。否则，验证方执行以下步骤。

7.4.5.3 准备待验证消息

验证方从已签名消息中提取M，并将消息分为两部分M_1和M_2，使得M_2为空且M_1等于M。

7.4.5.4 恢复赋值

赋值为$T = (T_1, T_2)$，其中$T_1 = [-Y^{-1}]$，$T_2 = [Y^{-1}R]P$。但是，验证者不需要计算该赋值。

7.4.5.5 重计算预签名

验证方重新计算预签名值[见公式(108)]：

$$\Pi' = < S, [Y]Q + V > * < P, V >^{R} \quad (108)$$

注：$< P, V >$可以预先计算。

7.4.5.6 重计算见证值

验证方重新计算见证值[见公式(109)]：

$$R' = h_2(M_1 || FE2BS(r, \Pi_{a'}) || FE2BS(r, \Pi_{b'}), q) \quad (109)$$

对于更高扩张度的域，杂凑值中将出现更多项。例如，对于扩张度 3，$\Pi' = (\Pi_{a'}, \Pi_{b'}, \Pi_{c'})$，$h_2$的将为公式(110)：

$$M_1 || FE2BS(r, \Pi_{a'}) || FE2BS(r, \Pi_{b'}) || FE2BS(r, \Pi_{c'}) \quad (110)$$

7.4.5.7 验证见证值

验证方检查$R' = R$是否成立。若成立，则签名通过验证；否则无效。

附 录 A

（规范性）

对象标识符

附录 A 列出了本文件中指定的数字签名机制所分配的对象标识符，并定义了算法参数结构。

```
DigitalSignatureWithAppendixDL {
    iso(1) standard(0) digital-signature-with-appendix (14888) part3(3)
        asn1-module(1) discrete-logarithm-based-mechanisms(0) }
DEFINITIONS EXPLICIT TAGS ::= BEGIN
-- EXPORTS All; --
IMPORTS
    HashFunctions
        FROM DedicatedHashFunctions {
            iso(1) standard(0) encryption-algorithms(10118) part3(3)
            asn1-module(1)
                dedicated-hash-functions(0) } ;
OID ::= OBJECT IDENTIFIER --  别名
–  同义词  –
id-dswa-dl OID ::= {
    iso(1) standard(0) digital-signature-with-appendix(14888) part3(3)
        algorithm(0) }
–  赋值  –
id-dswa-dl-DSA OID ::= { iso(1) member-body(2) us(840) ansi-x9-57(10040)
x9cm(4) dsa(1) }
id-dswa-dl-KCDSA OID ::= { id-dswa-dl kcdsa(2) }
id-dswa-dl-PVS OID ::= { id-dswa-dl pvs(3) }
id-dswa-dl-EC-DSA OID ::= { iso(1) member-body(2) us(840) ansi-x9-62(10045)
signatures(4) ecdsa-with-Recommended(2) }
id-dswa-dl-EC-KCDSA OID ::= { id-dswa-dl ec-kcdsa(5) }
id-dswa-dl-EC-GDSA OID ::= { id-dswa-dl ec-gdsa(6) }
id-dswa-dl-IBS-1 OID ::= { id-dswa-dl ibs-1(7) }
id-dswa-dl-IBS-2 OID ::= { id-dswa-dl ibs-2(8) }
id-dswa-dl-EC-RDSA OID ::= { id-dswa-dl ec-rdsa(9) }
id-dswa-dl-SDSA OID ::= { id-dswa-dl sdsa(10) }
id-dswa-dl-EC-SDSA OID ::= { id-dswa-dl ec-sdsa(11) }
id-dswa-dl-EC-FSDSA OID ::= { id-dswa-dl ec-fsdsa(12) }
id-dswa-dl-EC-SDSA-opt OID ::= { id-dswa-dl ec-sdsa-opt(13) }
id-dswa-dl-SM2 OID ::= { id-dswa-dl sm2(14) }
id-dswa-dl-Chinese-IBS OID ::= { id-dswa-dl chinese-IBS(15) }
DigitalSignatureWithAppendix ::= SEQUENCE {
    algorithm ALGORITHM.&id(DSAlgorithms),
    parameters ALGORITHM.&Type(DSAlgorithms@algorithm) OPTIONAL
}
```

```
DSAlgorithms ALGORITHM ::= {
    dswa-dl-DSA |
    dswa-dl-KCDSA |
    dswa-dl-PVS |
    dswa-dl-EC-DSA |
    dswa-dl-EC-KCDSA |
    dswa-dl-GDSA |
    dswa-dl-IBS-1 |
    dswa-dl-IBS-2 |
    dswa-dl EC-RDSA |
    dswa-dl SDSA |
    dswa-dl EC-SDSA |
    dswa-dl EC-FSDSA |
    dswa-dl EC-SDSA-opt,
    … – 预期包含更多算法 –
}
    dswa-dl-DSA ALGORITHM ::= {
        OID id-dswa-dl-DSA PARMS NullParms
    }
    dswa-dl-KCDSA ALGORITHM ::= {
        OID id-dswa-dl-KCDSA PARMS HashFunctions
    }
    dswa-dl-PVS ALGORITHM ::= {
        OID id-dswa-dl-PVS PARMS HashFunctions
    }
    dswa-dl-EC-DSA ALGORITHM ::= {
        OID id-dswa-dl-EC-DSA PARMS NullParms
    }
    dswa-dl-EC-KCDSA ALGORITHM ::= {
        OID id-dswa-dl-EC-KCDSA PARMS HashFunctions
    }
    dswa-dl-EC-GDSA ALGORITHM ::= {
        OID id-dswa-dl-EC-GDSA PARMS HashFunctions
    }
    dswa-dl-IBS-1 ALGORITHM ::= {
        OID id-dswa-dl-IBS-1 PARMS HashFunctions
    }
    dswa-dl-IBS-2 ALGORITHM ::= {
        OID id-dswa-dl-IBS-2 PARMS HashFunctions
    }
    dswa-dl-EC-RDSA ALGORITHM ::= {
        OID id-dswa-dl-EC-RDSA PARMS HashFunctions
    }
    dswa-dl-SDSA ALGORITHM ::= {
```

OID id-dswa-dl-SDSA PARMS HashFunctions

}

dswa-dl-EC-SDSA ALGORITHM ::= {

OID id-dswa-dl-EC-SDSA PARMS HashFunctions

}

dswa-dl-EC-FSDSA ALGORITHM ::= {

OID id-dswa-dl-EC-FSDSA PARMS HashFunctions

}

dswa-dl-EC-SDSA-opt ALGORITHM ::= {

OID id-dswa-dl-EC-SDSA-opt PARMS HashFunctions

}

NullParms ::= NULL

– 密码算法标识 –

ALGORITHM ::= CLASS {

&id 对象标识符唯一,

&Type 可选

}

具有语法 { OID &id [PARMS &Type] }

END -- DigitalSignatureWithAppendixDL --

注 1：KCAC.TG.OID 中提出的 KCDSA 的替代 OID 如下：

{iso(1) member-body(2) korea(410) kisa(20004) npki-alg(1) kcdsa1(21)}

——KCDSA：

{iso(1) member-body(2) korea(410) kisa(20004) npki-alg(1) kcdsa1WithHAS160(22)}

——使用 HAS160 的 KCDSA，其中 HAS160 是韩国杂凑标准算法：

{iso(1) member-body(2) korea(410) kisa(20004) npki-alg(1) kcdsa1WithSHA1(23)}

——使用 SHA1 的 KCDSA：

注 2：TTAS.KO-12.0015 中提出的 EC-KCDSA 与 HAS160 的替代 OID 为：

{iso(1) member-body(2) korea(410) kisa(20004) npki-alg(1) ecc(100) signature(4) eckcdsa-with-HAS160(1)}。

附 录 B

（规范性）

转换函数（I）

B.1 从域元素到整数的转换：FE2I(r,x)

输入值：

——r为一个素数或素数的幂；

——x为 Galois 域$GF(r)$中的一个元素。

假设：

——当$r = p$，其中p是一个奇素数时：$x \in GF(p)$表示为在$\{0,1,\ldots,p-1\}$中的一个整数。

——当$r = p^m$，其中p为奇素数，m为大于 1 的整数时：

$x \in GF(p^m)$表示为长度为m的p进制字符串；

$x = x_{m-1}x_{m-2}\ldots x_0$，其中$x_i \in \{0,1,\ldots,p-1\}$，对于$0 \le i < m$。

——当$r = 2$时：$x \in GF(2)$表示为 0 或 1。

——当$r = 2^m$，其中m为大于 1 的整数时：

$x \in GF(2^m)$表示为长度为m的二进制字符串；

$x = x_{m-1}x_{m-2}\ldots x_0$，其中$x_i \in \{0,1\}$，对于$0 \le i < m$。

输出值：

——当$r = p$，其中p是一个奇素数时：$FE2I(r,x) = x \in \{0,1,\ldots,p-1\}$；

——当$r = p^m$，其中p为奇素数，m为大于 1 的整数，且x表示为p进制字符串$x_{m-1}x_{m-2}\ldots x_0$时：

$FE2I(r,x) = p^{m-1}x_{m-1} + p^{m-2}x_{m-2} + \ldots + x_0 \in \{0,1,\ldots,p^{m-1}\}$。

——当$r = 2$时：$FE2I(r,x) = x \in \{0,1\}$。

——当$r = 2^m$，其中m为大于 1 的整数，且x表示为二进制字符串$x_{m-1}x_{m-2}\ldots x_0$时：

$FE2I(r,x) = 2^{m-1}x_{m-1} + 2^{m-2}x_{m-2} + \ldots + x_0 \in \{0,1,\ldots,2^{m-1}\}$。

B.2 从整数到域元素的转换：I2FE(r,x)

输入值：

——r为一个素数或素数的幂。

——x为在$\{0,1,\ldots,p-1\}$中的一个整数。

假设：

——当$r = p$，其中p是一个奇素数时：$GF(p)$的元素表示为在$\{0,1,\ldots,p-1\}$中的一个整数。

——当$r = p^m$，其中p为奇素数，m为大于 1 的整数时：$GF(p^m)$的元素表示为长度为m的p进制字符串。

——当$r = 2$时：$GF(2)$的元素表示为在$\{0,1\}$中的一个整数。

——当$r = 2^m$，其中m为大于 1 的整数时：$GF(2^m)$的元素表示为长度为m的二进制字符串。

输出值：

——当$r = p$是一个奇素数，$I2FE(r,x) = x \in \{0,1,\ldots,p-1\}$。

——当$r = p^m$，其中p为奇素数，m为大于 1 的整数时：$I2FE(r,x) = x_{m-1}x_{m-2}\ldots x_0$。

由m个分量组成的p元字符串（表示x的基p展开，必要时可前置零以达到所需长度）可按下述方式计算。

a: = x;
i: = 0;
While(i < m)do{
b: = $\lfloor a/p \rfloor$;
xi: = $a - (p)(b)$;
a: = b;
i: = $i + 1$}

——当$r = 2$时，$I2FE(r, x) = x \in \{0,1\}$。

——当$r = 2^m$，其中 m 为大于 1 的整数时：$I2FE(r, x) = x_{m-1}x_{m-2}\ldots x_0$，

一个由m个分量组成的二进制字符串（表示x的基 2 展开，必要时加前置零以达到所需长度）可按下述方式计算。

a: = x;
i: = 0;
$While(i < m)do\{$
b: = $\lfloor a/2 \rfloor$;
x: = $a - (2)(b)$;
a: = b;
i: = $i + 1\}$

B.3 从域元素到二进制字符串的转换：FE2BS(r, x)

输入值：

——r为一个素数或素数的幂。

——x为 Galois 域$GF(r)$中的一个元素。

假设：

——当$r = p$时，其中p是一个奇素数，$x \in GF(p)$表示为在$\{0,1,\ldots,p-1\}$中的一个整数。

——当$r = p^m$时，其中p为奇素数，m为大于 1 的整数：

$x \in GF(p^m)$表示为长度为m的p进制字符串；

$x = x_{m-1}x_{m-2}\ldots x_0$，其中$x_i \in \{0,1\}$，对于$0 \le i < m$。

——当$r = 2$时，$x \in GF(2)$表示为在$\{0,1\}$中的一个整数

——当$r = 2^m$时，其中m为大于 1 的整数：$x \in GF(2^m)$表示为长度为m的二进制字符串；

$x = x_{m-1}x_{m-2}\ldots x_0$，其中$x_i \in \{0,1\}$，对于$0 \le i < m$

输出值：

——$FE2BS(r, x) = I2BS(g, FE2I(r, x))$,其中$g = 8\lceil log256(r)\rceil$ 。

B.4 从二进制字符串到整数的转换：BS2I(g, x)

输入值：

——g为一个正整数，表示输入字符串的长度。

——$x = x_{g-1}x_{g-2}\ldots x_0$，是长度为$g$的二进制字符串。

输出值：

——$BS2I(g, x) = 2^{g-1}x_{g-1} + 2^{g-2}x_{g-2} + \ldots + x_0 \in \{0,1,\ldots,2^{g-1}\}$。

B.5 从整数到二进制字符串的转换：I2BS(g, x)

输入值：

——g为一个正整数，表示输出字符串的长度。

——x为在 $0,1,\ldots,2^{g-1}$中的一个整数。

输出值：

——$I2BS(g,x)=x_{g-1}x_{g-2}\ldots x_0$。一个二进制字符串，其$g$个分量（表示$x$的二进制展开，如有必要，前面补零以达到所需长度）可按如下方式计算。

$a:=x;$

$i:=0;$

$While(i<g)do\{$

$b:=\lfloor a/2\rfloor;$

$xi:=a-(2)(b);$

$a:=b;$

$i:=i+1\}$

B.6 整数与八位位组字符串之间的转换：I2OS(h,x)和 OS2I(h,M)

I2OS$(\boldsymbol{h},\boldsymbol{x})$：

输入值：

——h为一个正整数，表示输出八位位组字符串的长度。

——x为在 $0,1,\ldots,256^{h-1}$中的一个整数。

输出值：

——计算整数串$x_{h-1}x_{h-2}\ldots x_0$，其中对于 $0\leq i<h$，$x_i\in\{0,1,\ldots,255\}$，表示一个基数x的 256 进制展开，必要时用前导零填充以达到长度h。x_i值可计算如下。

$a:=x;$

$i:=0;$

$While(i<h)do\{$

$b:=\lfloor a/256\rfloor;$

$xi:=a-(256)(b);$

$a:=b;$

$i:=i+1\}$

—I2OS$(h,x)=M_{h-1}M_{h-2}\ldots M_0$，其中八位组$M_{\mathrm{i}}$等价于 8 位二进制串 I2BS$(8,x_{\mathrm{i}})$。

OS2I$(\mathbf{h},\mathbf{M})$:

输入值：

——h为一个正整数，表示输入八位组串的长度。

——$M=M_{h-1}M_{h-2}\ldots M_0$，是长度为$h$的八位组串。

假设：

——对于 $0\leq i<h$，M_{i}表示为 8 位二进制串。

输出值：

——计算整数串$x_{h-1}x_{h-2}\ldots x_0$，其中$x_{\mathrm{i}}=\mathrm{BS2I}(8,M_{\mathrm{i}})\in\{0,1,\ldots,255\}$，$0\leq i<h$。

——$\mathrm{OS2I}(h,x)=256^{h-1}x_{h-1}+256^{h-2}x_{h-2}+\ldots+x_0\in\{0,1,\ldots,256^{h-1}\}$。

附 录 C

（资料性）

转换函数（I I）

附录 C 规定了函数 I2P（整数到点转换），用于指定第 7 章中的两个基于身份的签名机制。

该函数被认为具有随机性和单向性，即将整数转换为点，使得该点在所选群中随机分布，并且根据当前知识，从该点恢复整数在计算上是不可行的。该函数也被 IEEE P1363 使用。然而，该函数尚无正式的安全性证明发布。因此，该函数被推荐为资料性。

给定一组椭圆曲线域参数(r, q, a_1, a_2)，函数 I2P 以整数u为输入，并在域$GF(r)$上的曲线E上生成一个阶为q的点T作为输出，即$T = I2P(u)$。在以下规范中，有限域元素之间的加法和乘法运算遵循 ISO/IEC 15946-1 中的规范，KDF1 运算遵循 ISO/IEC 18033-2 中的规范。

a) 设 v = BS2I(8 ⌈$log256(r)$⌉, KDF1H_2(I2OS(⌈$log256(u)$⌉, u), r 的以字节表示的长度))$mod\ r$。如果$v = 0$，输出“invalid”并停止。

b) 设$\lambda = u\ mod\ 2$。

c) 如果r是一个素数$(r = p)$，曲线E是$GF(p)$上定义的$Y^2 = X^3 + a_1X + a_2$，以以下方式计算点 T：

1) 令x取v的值。

2) 计算域元素$c = x^3 + a_1x + a_2\ mod\ p$。如果$c = 0$,输出“invalid”并停止。

3) 计算$c\ mod\ p$的平方根d(即，整数d（$0 < d < p$），$d^2 = c\ mod\ p$)或确定不存在这样的平方根。

i) 计算$\delta = C^{(p-1)/2}/2\ mod\ p$以确定是否存在平方根。如果$\delta = 1$,则$d$存在；否则，$d$不存在。

ii) 如果$\delta \neq 1$，计算$u = u + 1$，跳转到a）。

iii) 如果$\delta = 1$，寻找d。

在参考文献[4]和参考文献[23]中，给出了找到满足 $d^2 = c\ mod\ p$的两个域元素d的操作。为使结果唯一，如果该机制的应用对选择哪个值有特定要求，则遵循该要求。否则，推荐选择模p的最小绝对值。

iv) 设$y = (\text{I2FE}(r, p-1))^{\lambda} \times d$。

v) 设点$T = (x, y)$，计算$T = [\#E/q]T$，输出T。

d) 如果r是一个奇素数幂$(r = p^m, p > 2, m \geq 2)$，椭圆曲线$E$是$Y^2 = X^3 + a_1X + a_2$*(p=3)* 和在$GF(p^m)$上定义的$Y^2 = x^3 + a_1x + a_2$（当$p > 3$ 时），按以下方式计算点T：

1) 设$x = \text{I2FE}(r, v)$。

2) 如果$p = 3$，设在$GF(p^m)$上的$c = X^3 + a_1X^2 + a_2\ mod\ p$。如果$c = 0$，输出“invalid”并停止。

3) 如果$p > 3$，设在$GF(p^m)$上$c = X^3 + a_1X + a_2$。如果$c = 0$，输出“invalid”并停止。

4) 找到$GF(p^m)$上c的平方根d，d^2=c in GF(p^m)]或确定不存在这样的平方根。如果结果是不存在这样的平方根，则设$u = u + 1$，跳转到a）。

注 1：在参考文献[4]和[23]中给出了确定域元素平方根是否存在以及查找平方根的操作。为使结果唯一，如果该机制的应用对选择哪个值有特定要求，则遵循该要求。否则，比较结果的最大次数，并选择次数较小的 d。如果两个值的最大次数相同，则选择该次数绝对值系数较小的 d。如果次数和系数都相同，则比较次大次数，并选择该次数绝对值系数较小的 d。重复此过程直到选出唯一的 d。

5) 设$y = (I2FE(r, p-1))^{\lambda} \times d$。

6) 设$T = (x, y)$，计算$T = [\#E/q]T$，输出T。

e) 如果r是 2 的素数幂$(r = 2^m, m \geq 2)$，曲线E是$GF(2^m)$上定义的$Y^2 + XY = X^3 + a_1X^2 + a_2$，以以下方式计算点$T$：

1) 设 x = I2FE(r, v)。

2) 设$GF(2^m)$的$c = x + a_1 + a_2x^{(-2)}$。如果$c = 0$,输出“invalid”并停止。

3) 在$GF(2^m)$中找到一个满足$d^2+d \equiv c$ 的域元素d，或确定不存在这样的整数。如果结果是不存在这样的整数，则设$u = u + 1$ 并转到步骤a）。

注 2：在参考文献[4]和[23]中给出了确定是否存在域元素d使得$d^2+d=c$ 以及找到该元素的操作。为使结果唯一，如果该机制的应用对选择哪个值有特定要求，则遵循该要求。否则，比较结果的最大次数，并选择最大次数较小的d。如果两个值的最大次数相同，则比较次大次数，并选择次大次数较小的d。重复此过程直到选出唯一的d。

4) 设$y = (d + \text{I2FE}(r, \lambda)) \times x$。

5) 设点$T = (x, y)$，计算$T = [\#E/q]T$，输出T。

附 录 D

（规范性）

DSA 域参数的生成

D.1 素数p和q的生成

素数生成方案从使用适当的杂凑函数和用户提供的 SEED 开始，以构造一个在范围$2\beta-1<q<2\beta$内的素数q。一旦完成，使用相同的 SEED 值构造一个在范围$2^{\alpha-1}<X<2^{\alpha}$内的$X$。然后通过将$X$舍入为与$1\ mod\ 2q$同余的数来形成素数$p$，如下所述。整数与序列之间的转换函数在附录 B 中给出。令h为针对(α,β)对的适当杂凑函数，并令$m(=\gamma)$为其输出块的长度。令$\alpha-1=n*m+b$，其中b和n是整数，且$0\le b<m$。

步骤 1：选择任意一个长度至少为β比特的序列，并将其称为 SEED。令s为 SEED 的比特长度。

步骤 2：计算$U=h(SEED)\ mod\ 2^{\beta}$。

步骤 3：通过设置U的最高有效位（第 2β−1 位）和最低有效位为 1 来形成q。用布尔运算表示，$q=UOR2^{\beta-1}OR1$。注意，$2^{\beta-1}<q<2^{\beta}$。

步骤 4：使用稳健的素性测试算法检验q是否为素数（稳健的素性测试是指非素数通过测试的概率至多为$2^{-\beta/2}$）。

步骤 5：如果q不是素数，则转至步骤 1。

步骤 6：设$counter=0$和$offset=1$。

步骤 7：对于$k=0,\ldots,n$，令$V_k=h((SEED+offset+k)\ mod\ 2^{s})$

步骤 8：令W为整数$W=V_0+V_1*2^{m}+\ldots+V_{n-1}*2^{(n-1)*m}+(V_n \bmod 2^{b})*2^{n*m}$，令$X=W+2^{\alpha-1}$。

注：$0\le W<2^{\alpha-1}$，因此$2^{\alpha-1}\le X<2^{\alpha}$

步骤 9：令 c = $X \bmod 2q$，并设$p=X-(c-1)$。注意，$p\equiv 1 \bmod 2q$。

步骤 10：如果$p<2^{\alpha-1}$，跳转到步骤 13。

步骤 11：对p执行稳健的素性测试。

步骤 12：若p通过步骤 11 的测试，则转至步骤 15。

步骤 13：令 counter = counter + 1，$offset=offset+n+1$。

步骤 14：若 counter ≥ 4^{α}，则转至步骤 1；否则（即 counter < 4^{α}）转至步骤 7。

步骤 15：保存 counter 值，并可选地保存 SEED 值，用于证明p和q的正确生成。

注：本过程摘自参考文献[17]附录 A。

D.2 生成元 G 的生成

D.2.1 不可验证的 G 生成

当不需要验证G时，使用此方法确定生成元G。G的值由p和q确定。

步骤 1：$e=(p-1)/q$。

步骤 2：设置F为任一整数，满足 $1<F<p-1$，且F不同于之前尝试过的任何值。

步骤 3：$G=F^{e}\ mod\ p$。

步骤 4：如果$G=1$，跳转到步骤 2。

D.2.2 可验证的 G 生成

对于该方法，生成元G基于p、q、index 和 SEED 的值。index 是长度为 8 位的表示无符号整数的比特串。index 可用于从同一(p,q)对生成不同的G值。SEED 的值是 D.1 所述算法

中最终保存的值。令h为适用于(α,β)对的杂凑函数。注意，本方法支持为特定的p和q值生成多个G值。使用不同的G值可以通过为 index 提供不同的值来支持密钥分离。此处，“ggen”为 ASCII 字符串 0x6767656E，count 为 16 位计数器（即模2^{16}定义的无符号整数）。

步骤 1：$e = (p-1)/q$。

步骤 2：$count = 1$。

步骤 3：$U = SEED \,||\, \text{“}ggen\text{”} \,||\, index \,||\, count$。

步骤 4：$W = h(U)$。

步骤 5：$G = W^e \bmod p$。

步骤 6：若$G < 2$，则增加 count 并转至步骤 3。

附 录 E

（资料性）

Weil 配对与 Tate 配对

E.1 概述

Weil 配对和 Tate 配对均为椭圆曲线E上点对P、Q的函数$< P,Q >$。它们用于第 7 章规定的两种基于身份的机制。

令G_1和G_2表示两个素数阶q的群，其中G_1采用加法符号表示椭圆曲线E上的点群；G_2采用乘法符号表示有限域乘法群的子群。

配对是这两个群之间可计算的双线性映射。已有两种配对被研究用于密码学：Weil 配对[27][35]及其修改版本[11]，以及 Tate 配对[18][19]的修改版本。在本文件中，<>表示一般的双线性映射，即<>:$G_1 \times G_1 \to G_2$，可以是修改后的 Weil 配对或 Tate 配对。修改后的 Weil 配对和 Tate 配对具有以下两个性质：

——双线性：若 P、P_1、P_2、Q、Q_1、Q_2是素数阶 q 的循环群中的点，且 a 满足 $1 \le a \le q-1$：

$$< P_1 + P_2, Q >=< P_1, Q >*< P_2, Q > ;$$

$$< P, Q_1 + Q_2 >=< P, Q_1 >*< P, Q_2 > ;$$

$$< [a]P, Q >=< P, [a]Q >=< P, Q > a;$$

——非退化：若P是循环群中的非单位元点，则$< P, P >\neq 1$。

E.2 函数f、g和d

以下三个函数用于计算 Weil 配对和 Tate 配对。

——设E为椭圆曲线，其公式为$y^2 + a_1 * x * y + a_3 * y = x^3 + a_2 * x^2 + a_4 * x + a_6$。

——给定E上的三个有限点(x_0,y_0)、(x_1,y_1)、(u,v)，按如下定义函数$f((x_0,y_0)$、$(x_1,y_1),(u,v))$：

如果$(x_0,y_0)=0_E$,且$(x_1,y_1)=0_E$，则$f=1$

否则，如果$(x_0,y_0)=0_E$，则$f = u - x_1$

否则，如果$(x_1,y_1)=0_E$，则$f = u - x_0$

否则，如果$x_0 \neq x_1$，则$f = (x_1 - x_0) * v - (y_1 - y_0) * u - x_1 * y_0\ x_0 * y_1$

否则，如果$y_0 \neq y_1$，则$f = u - x_0$

否则$f = (a_1 * y_0 - 3 * {x_0}^2 - 2 * a_2 * x_0 - a_4) * (u - x_0) + (2 * y_0 + a_1 * x_0 + a_3) * (v - y_0) = -(v - y_0)^2 - (u - x_0) * (a_1 * (v - y_0) - (u - x_0) * (2 * x_0 + a_2 + u))$

——已知E上的点A、B、C，令$g(A,B,C) = f(A,B,C)/f(A+B,-(A+B),C)$。

注：取决于第 4 章中定义的值r和曲线E，上述函数f可能得到简化。以下是几个广泛使用的示例，但并未涵盖所有可能情况。若r为素数$(r = p)$且曲线E为定义在$GF(p)$上的$y^2 = x^3 + a_4 * x + a_6$，或若$r$为奇素数幂$(r = p^m, p > 3, m \ge 2)$且曲线$E$为定义在$GF(p^m)$上的$y^2 = x^3 + a_4 * x + a_6$，则函数$f$可写为$f = (-3 * {x_0}^2 - a_4) * (u - x_0) + 2 * y_0 * (v - y_0) = -(v - y_0)^2 + (u - x_0)^2 * (2 * x_0 + u)$。*如果*$r$是 2 的幂次方$(r = 2^m, m \ge 2)$且曲线$E$为定义在 GF($2^m$) 上的$y^2 + x * y = x^3 + a_2 * x^2 + a_6$，则函数$f$可写为当$f = (y_0 + {x_0}^2) * (u + x_0) + x_0 * (v + y_0) = (v + {y_0}^2) + (u + x_0) * (v + y_0 + (u + x_0) * (u + a_2))$。如果$r$是 3 的幂次方$(r = 3^m, m \ge 2)$且曲线$E$为定义在$GF(3^m)$，上的$y^2 = x^3 + a_2 * x^2 + a_6$，则函数$f$可写为$f = 2 * y_0 * (v - y_0) - 2 * a_2 * x_0 * (u - x_0)$。

——已知E上的两个点D和C以及一个整数$l > 2$，通过以下算法计算 Weil 函数$d(D,C,l)$。

- 设$A = D, f = 1$。令$l = (n_t, \ldots, n_0)$为 1 的比特表达$l = \sum i\ n_i\ 2^i$且$n_t \neq 0$。
- $for\ i = t-1, t-2, \ldots, 0\ do$\{
  $f = f * f * g(A, A, C)$;
  $A = A + A$;
  if ni $\neq$ 0 then\{
  $f = f * g(A, D, C)$;
  $A = A + D$;
  \}
  \}
- 设$d(D, C, l) = f$，输出$d(D, C, l)$。省略参数 l 时，$d(D, C, l)$记作 d(D, C)。

E.3 Weil 配对

设 *l>2* 为素数，且P和Q是E上的点，满足$[l]P = [l]Q = O_E$，则 Weil 配对$< P, Q >$可通过以下步骤计算：

——在E上选取一个随机点T（使得$0_E, Q, T, P + T$互不相同）；

——计算$< P, Q >= ((d(P, Q - T)/d(P, -T))/(d(Q, P + T)/d(Q, T)))$。

如果在配对计算过程中尝试除以零，则应使用新点 T 重新开始计算。

E.4 Tate 配对

设 *l>2* 为素数，且P和Q是E上的点，满足$[l]P = O_E$，则 Tate 配对$< P, Q >$可通过以下步骤计算：

——在E上选取一个随机点T；

——计算$< P, Q >= (d(P, Q - T)/d(P, -T))$。

如果在配对计算过程中尝试除以零，则宜使用新点T重新开始计算。

注：关于 Weil 和 Tate 配对实现的更多详细信息可在参考文献[8][21][30]中找到。

E.5 约化 Tate 配对

设$l > 2$ 为素数，且P和Q是E上的点，满足$[l]P = O$，则配对< P, Q >可通过以下步骤计算：

——在E上选取一个随机点T；

——计算$< P, Q >= (d(P, Q - T)/d(P, -T))^{(p^k - 1)/l}$;

式中：

k——扩域的维数（例如$k = 2$）；

p^k——扩域中元素的数量。

如果在配对计算过程中尝试除以零，则宜使用新点T重新开始计算。

注 1：关于配对实现的更多详细信息可在参考文献[4]和[23]中找到。

注 2：约化 Tate 配对用于附录 F.11 和 F.12 的数值示例。

附 录 F
（ 资 料 性 ）
数 值 示 例

F.1 概述

在附录*F*中，除非另有明确说明，所有数值均以十六进制表示法给出。

建议仅对遗留应用宜使用基于 SHA-1 和 RIPEMD-160 的数字签名。

F.2 DSA 机制

F.2.1 示例 1：2 048 位素数p，SHA-224

F.2.1.1 概述

所有数值生成的完整说明见参考文献[17]。本示例为$\alpha = 2048$ 且$\beta = 224$ 的 DSA 样本值。所有哈希运算（包括域参数的生成）均使用 SHA-224 执行。

F.2.1.2 参数

α = 2 048

β = 224

SEED = 0C088E11 2F88B186 90421876 5614496E C2AF9770 C71D0A56 87F489B6

F = 2

p = B4865EFC 44BFB4CB 7EE034F0 EAE8A72D 25897819 9BF9BA28 8462FD97
19F33272 C010A11B 33BCE4E8 481B6EC7 AB1229D9 FC7BEA43 8055907F
F1E28FAC 33716089 DCED277F 9036440A 887D4B22 CAC5BABD ECD6A1B3
A1731594 20371025 BAAB5F18 D5FDE928 CE4F5EE4 5352785F 20057782
2C20756E 171CBDD8 1CEB932A E0F29109 5CFFD9C2 3A07AC6B C2F5250B
B9F8E2E6 5AF85215 6E8EEBF8 31C098FB 010057BD 425132B8 0A46BB5C
E801E241 05058E58 091383F1 6F124894 FB6DE9CD 3BCC4C6E 64901743
AF8F47C3 5CC2177E B15ED172 B4969174 FE3F645A 9D3BEFC6 811A9074
BF702024 98E5E157 ECDBED3C 1FDF3C4F 00DAB43A CBA49802 79392E18
B515851F

q = B4D0963C 40D74138 69F42710 BBEF73CB C6C1C4E6 35C6B9F3 CF7A6255

Counter = 24

G = A92434D5 6752B028 CF11954E 0F3B1BED 8804EB74 8DEED793 E2932E80
8F37C34A 15444A06 9A8B17E5 4BF7FB82 7D6FE959 428BA0CC 1F3B2B8E
EA0A25A2 CAF73A0C 68C7DC48 093374A3 CD1F2250 8EF05038 9E8AE58C
E6A8AD50 2510B4CA C42528B7 BCA0993C C959C630 61D7BA3A 885E9C6D
CA6EAE44 E2D3C050 A236645F FBDE4BA6 1ECEB17B 941F85E9 C5234A28
FAD461DE 8B55F033 DB7E0CB4 DA5E115F FFCD416D 5A8BC9CD 9DAA6816
010841CC 9F416A6F E109A40A 823874F0 EDD92F45 738918AC 0CB925E7
AB8E692A 9336DB36 697E6C75 5B0243CA EBB61A38 79EABAF6 AC53F166
2740D6ED 3E3DB9BF A629390A 6A517FB0 B50D02E2 57178145 AF964626
57ABA465

F.2.1.3 签名密钥和验证密钥

X = A279D0A3 A4243A2B 16909C9E 0BBFEC32 0589E4DF 1BDDAE72 3BA7353B

Y = 31246FA1 CB8D1430 BDCDEBF0 5BB8C967 D24E6728 BA5C900C 50852741
3AFD496A F12EA9CC D80D8916 62A7B9B3 C2023212 08943D85 5D7EA110
B9512D1B 9E4AABAB 72B99005 25127129 EAB2CC8E 66B6E09C 49341ABF
184B2733 9114E39E FED6B90B 8D7BA182 3E3512D3 EB82F720 76C2815D
A642DE61 D808DCF0 22A76077 1E22AA42 26997E41 EA142BAD BFD00011
F7D27677 08A0313E 42255286 0D184F18 C4890ED3 A6CE8134 E1647DDC
B292B5FD 5C5ED61C 1BF9567A E1E40CC5 F85F5B7D 1A09AAA1 08CFCFE2
469360A9 48F61B4D 1CDCA791 1BB64070 94D9A78B A34ED943 97057791
DFC56691 1B4F7DD9 61A7EBB8 74923C59 2458D43D F171CB81 698AB7EE
2E9B92E6

F.2.1.4 每条消息数据

M = “abc”的 ASCII 形式 = 61 62 63

K = 2973C724 7F9BD6DB 3C08CD7A 1DA427DF 6780A7DD F3E09362 E8BA1293

$h(M)$ = 23097D22 3405D822 8642A477 BDA255B3 2AADBCE4 BDA0B3F7 E36C9DA7

F.2.1.5 签名

R = 1DFAAA6F 87DA6148 6529A2F3 4EBC7D89 3D42F405 F8DCBB33 93CC1A00

S = 4A3E6377 D09A4CD6 67BA9F9C E3982EE9 C1AA6E90 70F7C2F7 0EA23173

F.2.1.6 验证

R' = 1DFAAA6F 87DA6148 6529A2F3 4EBC7D89 3D42F405 F8DCBB33 93CC1A00

F.2.2 示例 2：3 072 位素数p，SHA-256

F.2.2.1 概述

本示例为 DSA 在$\alpha = 3072$ 且$\beta = 256$ 时的样本值。所有哈希运算（包括域参数的生成）均采用 SHA-256。

F.2.2.2 参数

α = 3 072

β = 256

$SEED$ = 193AFCA7 C1E77B3C 1ECC618C 81322E47 B8B8B997 C9C83515 C59CC446
C2D9BD47

F = 2

p = 90066455 B5CFC38F 9CAA4A48 B4281F29 2C260FEE F01FD610 37E56258
A7795A1C 7AD46076 982CE6BB 956936C6 AB4DCFE0 5E678458 6940CA54
4B9B2140 E1EB523F 009D20A7 E7880E4E 5BFA690F 1B9004A2 7811CD99
04AF7042 0EEFD6EA 11EF7DA1 29F58835 FF56B89F AA637BC9 AC2EFAAB
90340222 9F491D8D 3485261C D068699B 6BA58A1D DBBEF6DB 51E8FE34
E8A78E54 2D7BA351 C21EA8D8 F1D29F5D 5D159394 87E27F44 16B0CA63

2C59EFD1 B1EB6651 1A5A0FBF 615B766C 5862D0BD 8A3FE7A0 E0DA0FB2
FE1FCB19 E8F9996A 8EA0FCCD E5381752 38FC8B0E E6F29AF7 F642773E
BE8CD540 2415A014 51A84047 6B2FCEB0 E388D30D 4B376C37 FE401C2A
2C2F941D AD179C54 0C1C8CE0 30D460C4 D983BE9A B0B20F69 144C1AE1
3F9383EA 1C08504F B0BF3215 03EFE434 88310DD8 DC77EC5B 8349B8BF
E97C2C56 0EA878DE 87C11E3D 597F1FEA 742D73EE C7F37BE4 3949EF1A
0D15C3F3 E3FC0A83 35617055 AC91328E C22B50FC 15B941D3 D1624CD8
8BC25F3E 941FDDC6 20068958 1BFEC416 B4B2CB73

q = CFA0478A 54717B08 CE64805B 76E5B142 49A77A48 38469DF7 F7DC987E
FCCFB11D

Counter = 20

G = 5E5CBA99 2E0A680D 885EB903 AEA78E4A 45A46910 3D448EDE 3B7ACCC5
4D521E37 F84A4BDD 5B06B097 0CC2D2BB B715F7B8 2846F9A0 C393914C
792E6A92 3E2117AB 805276A9 75AADB52 61D91673 EA9AAFFE ECBFA618
3DFCB5D3 B7332AA1 9275AFA1 F8EC0B60 FB6F66CC 23AE4870 791D5982
AAD1AA94 85FD8F4A 60126FEB 2CF05DB8 A7F0F09B 3397F393 7F2E90B9
E5B9C9B6 EFEF642B C48351C4 6FB171B9 BFA9EF17 A961CE96 C7E7A7CC
3D3D03DF AD1078BA 21DA4251 98F07D24 81622BCE 45969D9C 4D6063D7
2AB7A0F0 8B2F49A7 CC6AF335 E08C4720 E31476B6 7299E231 F8BD90B3
9AC3AE3B E0C6B6CA CEF8289A 2E2873D5 8E51E029 CAFBD55E 6841489A
B66B5B4B 9BA6E2F7 84660896 AFF387D9 2844CCB8 B6947549 6DE19DA2
E58259B0 90489AC8 E62363CD F82CFD8E F2A427AB CD65750B 506F56DD
E3B98856 7A88126B 914D7828 E2B63A6D 7ED0747E C59E0E0A 23CE7D8A
74C1D2C2 A7AFB6A2 9799620F 00E11C33 787F7DED 3B30E1A2 2D09F1FB
DA1ABBBF BF25CAE0 5A13F812 E34563F9 9410E73B

F.2.2.3 签名密钥与验证密钥

X = 3ABC1587 297CE7B9 EA1AD665 1CF2BC4D 7F92ED25 CABC8553 F567D1B4
0EBB8764

Y = 8B891C86 92D3DE87 5879390F 2698B26F BECCA6B0 75535DCE 6B0C8625
77F9FA0D EF6074E7 A7624121 224A5958 96ABD4CD A56B2CEF B942E025
D2A4282F FAA98A48 CDB47E1A 6FCB5CFB 393EF35A F9DF9131 02BB303C
2B5C36C3 F8FC04ED 7B8B69FE FE0CF3E1 FC05CFA7 13B3435B 2656E913
BA8874AE A9F93600 6AEB448B CD005D18 EC3562A3 3D04CF25 C8D3D698
44343442 FA3DB7DE 618C5E2D A064573E 61E6D558 1BFB694A 23AC87FD
5B52D62E 954E1376 DB8DDB52 4FFC0D46 9DF97879 2EE44173 8E5DB05A
7DC43E94 C11A2E7A 4FBE3830 71FA36D2 A7EC8A93 88FE1C4F 79888A99
D3B61056 97C2556B 79BB4D7E 781CEBB3 D4866AD8 25A5E830 84607228
9FDBC941 FA679CA8 2F5F78B7 461B2404 DB883D21 5F4E0676 CF549395
0AC55916 97BFEA8D 1EE6EC01 6B89BA51 CAFB5F9C 84C989FA 117375E9
4578F28B 3ABC1587 297CE7B9 E0B34CE0 545DA462 66FD77F6 2D8F2CEE
92AB7701 2AFEBC11 008985A8 21CD2D97 8C7E6FE7 499D1AAF 8DE632C2
1BB48CA5 CBF9F310 98FD3FD3 854C49A6 5D920174 4AACE540 354974F9

F.2.2.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = A6902C1E 6E3943C5 62806158 8A8B007B CCEA91DB F1291548 3F04B24A
B0678BEE

$h(M)$ = BA7816BF 8F01CFEA 414140DE 5DAE2223 B00361A3 96177A9C B410FF61
F20015AD

F.2.2.5 签名

R = 5F184E64 5A38BE8F B4A6871B 6503A9D1 2924C7AB E04B7141 0066C2EC
A6E3BE3E

S = 91EB0C7B A3D4B9B6 0B825C3D 9F2CADA8 A2C9D772 3267B033 CBCDCF88
03DB9C18

F.2.2.6 验证

R' = 5F184E64 5A38BE8F B4A6871B 6503A9D1 2924C7AB E04B7141 0066C2EC
A6E3BE3E

F.3 KCDSA 机制

F.3.1 示例 1：2 048 位素数p，224 位素数q，SHA-224

F.3.1.1 概述

本示例使用 SHA-224 作为哈希函数h。哈希码即为 SHA-224 的输出值。

F.3.1.2 参数

l = 200 （即十进制的 512）

α = 2 048

β = 224

p = 8DA8C1B5 C95D11BE 46661DF5 8C9F803E B729B800 DD92751B
3A4F10C6 A5448E9F 3BC0E916 F042E399 B34AF9BE E582CCFC
3FF5000C FF235694 94351CFE A5529EA3 47DCF43F 302F5894
380709EA 2E1C416B 51A5CDFC 7593B18B 7E3788D5 1B9CC9AE
828B4F8F B06E0E90 57F7FA0F 93BB0397 031FE7D5 0A6828DA
0C1160A0 E66D4E5D 2A18AD17 A811E70B 14F4F431 1A028260
3233444F 98763C5A 1E829C76 4CF36ADB 56980BD4 C54BBE29
7E790228 4292D75C A3600FF4 59310B09 291CBEFB C721528A
13403B8B 93B711C3 03A2182B 6E6397E0 83380BF2 886AF3B9
AFCC9F50 55D8B713 6C0EBD08 C5CF0B38 888CD115 72787F6D
F384C97C 91B58C31 DEE5655E CBF3FA53

q = 864F1884 1EC103CD FD1BE7FE E54650F2 2A3BB997 537F32CC
79A51F53

G = 0E9BE1F8 7A414D16 7A9A5A96 8B079E4A D385A357 3EDB21AA
67A6F61C 0D00C14A 7A225044 B6E9EB03 68C1EB57 B24B45CD
854FD93C 1B2DFB0A 3EA302D2 367E4EC7 2F6E7EE8 EA7F8002
F7704E99 0B954F25 BADA8DA6 2BAEB6F0 6953C0C8 5104AD03

F36618F7 6C62F4EC F3480183 69850A56 17C999DB E68BA17D
5BC72556 74EF4839 22C6A3F9 9D3C3C6F 358896C4 E63C605E
E7DB16FC BD9BE354 E281F7FE 7813D054 27ED1912 B5C7653A
167B9434 9147EEAF 85CC9CE2 E81661F3 21512D5D 2C0580B0
3D1704EE F2317F45 185C8258 387E7EC9 79C04707 EF546241
2784AFE4 1A7B45C8 3B9CBE48 F9127CB4 400BE9E9 6AC5DE17
F2C9DEA3 5E3734E7 9B64673F 85681C4E

F.3.1.3 签名密钥与验证密钥

X = 2F1991C1 AF401872 8A5A431B 9B5459DF B16F6D25 6797FE57
0EC6BC65

Y = 04EDE5C6 7EA29297 A8CACB6B DE6F4666 AEA27D10 3DD1E9E9
582F76A2 F22B8B1B 32230BC5 8F06B768 F8102B49 FA1CAE5E
18921494 7F6239B6 C6CE7C9B C2D230E8 9A40BEE2 C33A8861
FD4F7D35 B788FE95 B2D5885D 8C8FAEA8 1C90BE4C EE2784E3
3577A71D 3B7F085D 71E9A1D4 7815C73F A087ACAA B9FCB565
5AC9570E 6852BE7C 9C0AECEA 8BD9AA75 A44FC314 7F733E90
6ADB0FD7 6D613561 B1DB364B BDC9AFD3 CE8F5F17 E3E71203
4A999350 8059FA52 441FA90D DFE9A0F2 A0B9192F E2220C08
1BD0C0F0 E07CB5F1 EE4FF405 23591F17 8A4FC7CB 5065F6A3
8216E9A0 99C205B2 9B8746D8 65E1AF6D 903E5A13 8004910B
70EB5B84 EED9760E A60578BF 08852898

F.3.1.4 逐消息数据

M = "This is a test message for KCDSA usage!"的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 74 65 73 74 20 6D
65 73 73 61 67 65 20 66 6F 72 20 4B 43 44 53 41
20 75 73 61 67 65 21

K = 49561994 FD2BAD5E 410CA1C1 5C3FD3F1 2E70263F 2820AD5C
566DED80

Y' = 1BD0C0F0 E07CB5F1 EE4FF405 23591F17 8A4FC7CB 5065F6A3
8216E9A0 99C205B2 9B8746D8 65E1AF6D 903E5A13 8004910B
70EB5B84 EED9760E A60578BF 08852898

$h(Y'\|M)$ = B3F921C1 8DDD06B6 09D02BD5 916F180E 5DFB19C8 9FEC063D
059A3575

V = 5E4E4BEC B42ED14C 1F00A98C D077A8C1 D65E6F5A 50D7ACD1
6AF7EC24

F.3.1.5 签名

R = EDB76A2D 39F3D7FA 16D08259 4118B0CF 8BA57692 CF3BAAEC
6F6DD951

S = 5260A2DF 2E923DE8 77B130AC 8B5E8B17 63973B88 D5D4627A
DFBACF52

F.3.1.6 验证

R' = EDB76A2D 39F3D7FA 16D08259 4118B0CF 8BA57692 CF3BAAEC
6F6DD951

F.3.2 示例 2：3 072 位素数p，256 位素数q，SHA-256

F.3.2.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。

F.3.2.2 参数

l = 200 （即十进制的 512）

α = 3 072

β = 256

p = CBAEACE3 677E98AD B2E49C00 2B8B0F43 4143B466 515839BF
813B097D 2D1EE681 5008C27A 3415BC22 31609874 5E5844F3
3ECC8887 C16DFB1C FB77DC4C 3F3571CC EEFD4291 8F6C48C3
702AB6EF 0919B7E8 402FC89B 35D09A0E 5040E309 1EE4674B
E891933C 1007E017 EDD40818 7E4114B6 BE5548D7 8DB58B84
8475A422 62D7EB79 5F08D161 1055EFEA 8A6AEB20 EB0F1C22
F002A2E8 195BCBBA 830B8461 3531BDD9 EC71E5A9 7A9DCCC6
5D6117B8 5D0CA66C 3FDAA347 6E97ADCD 05A1F490 2BD04B92
F400C42B A0C9940A 32600443 3B6D3001 28BF930F 484EAA63
02CD7A31 9EE5E561 A12A3625 594020C2 40DBA3BE BD8A4751
5841F198 EBE43218 2639616F 6A7F9BD7 434F0534 8F7F1DB3
115A9FEE BA984A2B 73784334 DE7737EE 3704535F CA2F4904
CB4AD58F 172F2648 E1D62D05 8539AC78 3D032D18 33D2B9AA
D96982C9 692E0DDB B6615508 83ED66F7 AA8BCE8F F0663A0A
DDA226C7 BD0E06DF C72594A3 87C676A3 CA06A300 62BE1D85
F23E3E02 C4D65E06 1B619B04 E83A318E C55ECA06 9EB85603

q = C2A8CAF4 87180079 66F2EC13 4EABA3CB B07F31A8 F2667ACB
5D9B872F A760A401

G = 17A1C167 AF836CC8 5149BE43 63F1BB4F 0010848F C9B678B4
E026F1F3 87133749 A4B1BBA4 C23252A4 C86F31E2 1E8ACACB
4E33AD89 B7C3D79A 5409268B FBA82B45 814E4352 0C09D631
613FA35D B9CAF18F 791C2729 A4B014BC 79A85A90 CD541037
119ECCDE 0778863F FCB9C259 31FCD33A 6706E5FE 1F495BB8
BCB3D0EE C9B6D5A9 373127A2 121E37D9 8A840330 258DBFCE
E7E06F81 5B69C16C 5D17289C 4CC37E71 9B856298 D4E1574E
4F4F8515 BAF9A850 D11DDA09 55BC30FA 5B16792D 673A3B1F
41512FC3 EB89452D 51509F97 4D878B48 2D2AD2ED 32BE1905
6F574504 2BFF804F B7482796 612B746F E8D70A83 8CC6F496
DD0FFC3D 95C1E0B1 98184D73 523656A0 6431BC52 5C2BC161
9729E8C0 88F6DF91 5645E060 922A4AF3 EDD63047 C7B6077C
667C07D8 8EB00F4C FE59D32E 5F545012 C566516B 7874FB3D

AED51403 31F29528 B30FC8B8 A9371C28 18017B09 53A84FFC
9FBFF84B 64BF0238 AA7E2AF2 ECADC15A 1C06DADC F1F2E7B1
240A5E64 5A6469C9 B002215D 9A91C2A4 ED2FB547 A942D777

F.3.2.3 签名密钥与验证密钥

X = 7C28569A 94B46FA7 45C8D306 AD7DC189 96CE046E EBE04383
8391C232 078DB05A

Y = 2574E10E 806F1C42 58F7CF8F A4A6CF2B EB177DBE 60E4EC17
DF21DCDB A72073F6 5565506D A3DF98D5 A6C8EEE6 1B6B5D88
B98C47C2 B2F6FC6F 504FA4FB C7F411E2 3EAA3B18 7A353DAE
D41533A9 558AB932 0A154CAE CC544E43 0008889A 2C899373
EC75A24C FF26247C F297D293 747ECC05 B3483647 A87BCBB8
D4500092 09F5E449 A00A659B 637CE139 CF6487AC A70F9C00
CB670C7F 3B95BFD7 CF236A0A 6F3C93BE 8D9CF591 C9D30686
9415B1AA 97264B90 4167850A 4794C780 BE4527DF FEB67BE6
E66786C5 CCE0378C CB49920D 855558F4 DAC4C42F 92DD229B
483B2257 DB0CE35D C737F980 1A261A02 BDF718C2 FD4D69C5
2E0D9712 B42C4897 BAE7C684 D3D35BC5 726CE899 2696B044
D722AFBA 78EFA858 C4D10F19 72112CE8 FFD39792 49BF14E4
9D8E0D9A CB1B0A9C A90D0551 1803845D 7C670BCF 1B066497
A7743B08 A219E764 EA0A3A2A 617661C1 6A372FE0 58B547A2
8B626ECF 442222E1 8EEF487C C101DBFB 715BC33A B85928EC
F0BD4DEA 30F250A6 A5C86178 83EA0F87 3E7A4651 98C4644B

F.3.2.4 逐消息数据

M = "This is a test message for KCDSA usage!"的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 74 65 73 74 20 6D
65 73 73 61 67 65 20 66 6F 72 20 4B 43 44 53 41
20 75 73 61 67 65 21

K = 83F3008F CEBAE57E C7A64A3A F7EE6EE1 9CC197A6 D5EBA3A5
B3EF79B2 F8F3DD53

Y' = EA0A3A2A 617661C1 6A372FE0 58B547A2 8B626ECF 442222E1
8EEF487C C101DBFB 715BC33A B85928EC F0BD4DEA 30F250A6
A5C86178 83EA0F87 3E7A4651 98C4644B

$h(Y'\|M)$ = 4D4F2A9B 83446B62 F571A669 FACB2D30 7ADE18DE 1A3FFB87
649ABA4E 606A0751

V = 1935B399 849AB60F 0AE62FAD 82B281E9 1A098A8F 51E6E7D6
BA581801 F02604A0

F.3.2.5 签名

R = 547A9902 07DEDD6D FF9789C4 7879ACD9 60D79251 4BD91C51
DEC2A24F 904C03F1

S = 1668797B 26641E72 94AA68D3 8562EAE3 CAA842D0 F446949C
4268AE3D 0392434F

F.3.2.6 验证

R' = 547A9902 07DEDD6D FF9789C4 7879ACD9 60D79251 4BD91C51
DEC2A24F 904C03F1

F.3.3 示例 3：2 048 位素数p，224 位素数q，SHA-256

F.3.3.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。

F.3.3.2 参数

l = 200 （即十进制的 512）

α = 2 048

β = 224

p = 8DA8C1B5 C95D11BE 46661DF5 8C9F803E B729B800 DD92751B
3A4F10C6 A5448E9F 3BC0E916 F042E399 B34AF9BE E582CCFC
3FF5000C FF235694 94351CFE A5529EA3 47DCF43F 302F5894
380709EA 2E1C416B 51A5CDFC 7593B18B 7E3788D5 1B9CC9AE
828B4F8F B06E0E90 57F7FA0F 93BB0397 031FE7D5 0A6828DA
0C1160A0 E66D4E5D 2A18AD17 A811E70B 14F4F431 1A028260
3233444F 98763C5A 1E829C76 4CF36ADB 56980BD4 C54BBE29
7E790228 4292D75C A3600FF4 59310B09 291CBEFB C721528A
13403B8B 93B711C3 03A2182B 6E6397E0 83380BF2 886AF3B9
AFCC9F50 55D8B713 6C0EBD08 C5CF0B38 888CD115 72787F6D
F384C97C 91B58C31 DEE5655E CBF3FA53

q = 864F1884 1EC103CD FD1BE7FE E54650F2 2A3BB997 537F32CC
79A51F53

G = 0E9BE1F8 7A414D16 7A9A5A96 8B079E4A D385A357 3EDB21AA
67A6F61C 0D00C14A 7A225044 B6E9EB03 68C1EB57 B24B45CD
854FD93C 1B2DFB0A 3EA302D2 367E4EC7 2F6E7EE8 EA7F8002
F7704E99 0B954F25 BADA8DA6 2BAEB6F0 6953C0C8 5104AD03
F36618F7 6C62F4EC F3480183 69850A56 17C999DB E68BA17D
5BC72556 74EF4839 22C6A3F9 9D3C3C6F 358896C4 E63C605E
E7DB16FC BD9BE354 E281F7FE 7813D054 27ED1912 B5C7653A
167B9434 9147EEAF 85CC9CE2 E81661F3 21512D5D 2C0580B0
3D1704EE F2317F45 185C8258 387E7EC9 79C04707 EF546241
2784AFE4 1A7B45C8 3B9CBE48 F9127CB4 400BE9E9 6AC5DE17
F2C9DEA3 5E3734E7 9B64673F 85681C4E

F.3.3.3 签名密钥与验证密钥

X = 2F1991C1 AF401872 8A5A431B 9B5459DF B16F6D25 6797FE57
0EC6BC65

Y = 04EDE5C6 7EA29297 A8CACB6B DE6F4666 AEA27D10 3DD1E9E9
582F76A2 F22B8B1B 32230BC5 8F06B768 F8102B49 FA1CAE5E
18921494 7F6239B6 C6CE7C9B C2D230E8 9A40BEE2 C33A8861
FD4F7D35 B788FE95 B2D5885D 8C8FAEA8 1C90BE4C EE2784E3

3577A71D 3B7F085D 71E9A1D4 7815C73F A087ACAA B9FCB565
5AC9570E 6852BE7C 9C0AECEA 8BD9AA75 A44FC314 7F733E90
6ADB0FD7 6D613561 B1DB364B BDC9AFD3 CE8F5F17 E3E71203
4A999350 8059FA52 441FA90D DFE9A0F2 A0B9192F E2220C08
1BD0C0F0 E07CB5F1 EE4FF405 23591F17 8A4FC7CB 5065F6A3
8216E9A0 99C205B2 9B8746D8 65E1AF6D 903E5A13 8004910B
70EB5B84 EED9760E A60578BF 08852898

F.3.3.4 逐消息数据

M = "This is a test message for KCDSA usage!"的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 74 65 73 74 20 6D
65 73 73 61 67 65 20 66 6F 72 20 4B 43 44 53 41
20 75 73 61 67 65 21

K = 49561994 FD2BAD5E 410CA1C1 5C3FD3F1 2E70263F 2820AD5C
566DED80

Y' = 1BD0C0F0 E07CB5F1 EE4FF405 23591F17 8A4FC7CB 5065F6A3
8216E9A0 99C205B2 9B8746D8 65E1AF6D 903E5A13 8004910B
70EB5B84 EED9760E A60578BF 08852898

$\mathrm{I2BS}(\beta, \mathrm{BS2I}(\gamma, h(Y'\|M_2)) \bmod 2^{\beta})$ =

894315A9 A68EF862 7D015AAE 4EBB41B3 FED88AAA 9614CC67
09DE2B47

V = 489142E8 F4A1ACE1 4F693AFD 632A6FB7 5A8392AD E61F8A26
3F665A1B

F.3.3.5 签名

R = C1D25741 522F5483 32686053 2D912E04 A45B1807 700B4641
36B8715C

S = 0AFAEA63 92942822 86B0F9E1 9EC1BC13 BFA29B54 5747F262
0DA3AFC1

F.3.3.6 验证

R' = C1D25741 522F5483 32686053 2D912E04 A45B1807 700B4641
36B8715C

F.3.4 示例 4：2 048 位素数p，224 位素数q，SHA-256（与 F.3.3 参数相同，但额外提供 F.3.4 以与参考文献[36]保持一致）

F.3.4.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。

F.3.4.2 参数

l = 200 （即十进制的 512）

α = 2 048

β = 224

p = C3159A30 CDBCC00C E2A99043 9634F7D3 FB16FEB1 2C579932
2C14F8B8 A0D9B98E 35F724BF E14C4AFC 475D78F9 3A83F8FB
4636A5DE F357BD6F B0C6245C AC4EF29C 8F7DA5E9 B39F3158
F4FD27C8 4088BCBB 6286D964 29C90E82 B7F31BF3 E76E93C6
8A3163CF B82370E2 75159D66 08F82601 013476D5 50B386CA
34736388 6DF337D7 A54DB7E9 8CC2DF0D 828C31EB C62F3BC2
3F070C89 9648E276 2B26FFED A9D88FFB F684C570 4937FEDC
03F60C10 5B69542E D40F910B 4C66FC09 1F5E1C12 47628ABC
E989B74A B0EF6F1A 14E2567F C083991E 1C846242 0BB8FBF9
B3F67B66 B02DE042 0A18D49A 6D4896D0 D1DDDBED 24EE1611
8090221F 9FE9A1E1 2194E0D2 B3C61C13

q = BB6A5C40 316BD80E 78246E92 AC9BF881 A9EB0CB9 6C7212EB
1E46AE0D

G = 487844C0 B67465B7 18F04DBD 453342B7 49076EE1 F4226F18
1DB282E1 C51B0F29 0DAE9601 AC73ED1F 1B25ADAD D50BFB42
1E8A09FA 07689A93 E5FB52A5 F8012956 B90641F8 45C4B7E4
45CAFE2E 3284775B DD70BCE4 0EF3274E 52CBC3D5 738DA7A8
61BC46C0 A9693AA8 7E0AAE62 BD371FA0 14FFC69F 3625D5A1
FBAAAC80 D81C78A5 9BADEAE5 FDFEA922 EBC330A1 37E7699A
2790E86B DB270C21 35EAB4E0 BCD28B77 13A8B241 1534C63F
2EDF4E00 5902F6CC 1A155C29 F3EAE17F 88ACB5C6 70F5CF19
A5A54E87 6692AB82 08C4A9EF 75A29E74 F08F92AC 1A38592D
46A2557C 3A18C06E D6529B40 BC5ECFF9 715329A2 C01B4245
874250ED 515537EE 7458F898 6FF920BC

F.3.4.3 签名密钥与验证密钥

X = B55D61EC 0114E020 EFC4C9BB 5F2F3D2E 38409E17 D3954174
6D94FF7C

Y = 0712496F CF76CE98 8BE97AC0 9F0DBBE6 2D58707A 767D608A
3301115D 479CC871 4CE3A10B EB152552 46C2623E FE50BFD2
5A83C355 551574E6 E3560E7B D1CD5E7E 8E1269A4 A6F1976C
84E8FE8E 32E55AED D548FCED CC92A6E4 E1BF2D1F 2AA30C0C
0A991C29 B2595029 F903B634 189AA70C FC429531 93016C1F
7BB6276D F3EBFAE7 C060B987 D89088A0 558FC132 27B86F7A
57DDE307 1CC022E0 39BE4B68 3858D782 F52AA730 49D508EF
994A5039 CAB5FAF2 89BDAC07 75EFBB51 EB4D5FF9 99B71D59
C4D833B5 D069202A 968F3AC3 5FA77BAF BDD9C096 0752C5DA
F783929D E2DAD916 F1159E75 A345445D 63C5B422 E0BCD2BA
D9379D14 43892ED5 D12F8285 3D51A705

F.3.4.4 逐消息数据

M = “This is a test message for KCDSA usage!”的 ASCII 形式

= 54 68 69 73 20 69 73 20 61 20 74 65 73 74 20 6D
65 73 73 61 67 65 20 66 6F 72 20 4B 43 44 53 41
20 75 73 61 67 65 21

K = A5C22F64 DDE15693 3AD15BCB 928D6A3B 5ACF0D7A 2302615C
E74CCAD6

Y' = C4D833B5 D069202A 968F3AC3 5FA77BAF BDD9C096 0752C5DA
F783929D E2DAD916 F1159E75 A345445D 63C5B422 E0BCD2BA
D9379D14 43892ED5 D12F8285 3D51A705

$\mathrm{I2BS}(\beta, \mathrm{BS2I}(\gamma, h(Y'\|M_2)) \bmod 2^{\beta})$ =
6B1908F3 D9E543EF 08C1C03A 185D8E22 45257F6C D1B454B7
4C2E342B

V = 38EE397D BD535F23 8BA66C32 49442FE9 FE002E63 300A9593
8EB7BDCB

F.3.4.5 签名

R = 53F7318E 64B61CCC 8367AC08 5119A1CB BB25510F E1BEC124
C29989E0

S = B750F725 1585204C 236E4204 884166A2 6C6CF08B D281167A
5EFADD52

F.3.4.6 验证

R' = 53F7318E 64B61CCC 8367AC08 5119A1CB BB25510F E1BEC124
C29989E0

F.4 Pointcheval/Vaudenay 机制

F.4.1 概述

本示例使用 2 048 位素数p及 SHA-224 作为哈希函数h。哈希码即为 SHA-224 的输出值。

F.4.2 示例 1：2 048 位素数p，SHA-224

F.4.2.1 参数

l = 200 （即十进制的 512）

F = 2

p = EEADFD4F 9CC4EFF7 F5B3F5D9 047399E1 00E6D01D 6DE30E40
39B25F76 083743F1 826919FE D5C750F5 BC6AD9C1 C0B9E229
695F306C 1AE9F631 7EC4AC94 74FD88F9 714798E6 2FD2D865
A2EE7BB4 E6C38A8F 2A4C1C30 1FEC7149 1356AD5A 87449AFF
F1AE412B 128ECF26 DB9DBD74 C91CBDCA 4ED4CE4F 43AC2770
E7B38648 6190B6A4 0211E61F F5F767C2 31F9FDEF FE999F47
BA2C2111 821C3EE5 BDD1BBC2 E4C89E5A 5C54DC16 0167A8D6
B0235223 44B45E72 AB2A54A7 A3787C59 6A38AA76 3589852C
50D67BFB 469278F0 7E6F5C03 2785FA22 2E635460 65E5E1D5
95BE7E94 F01C274C B3291116 40544989 38824CC9 55D802CE
76C0FD6E 33537111 78C33059 6F25CF05

q = C3E87841 9B4463CE 4FBA73B6 022498A0 EA51A6B8 050E79BB
A5FEF99D

G = BC376597 C582A659 072A5F1A 2839A1DF 594B0AE2 1262A447
A8A0F6CB 60596837 661C1734 DFC2BB0B 9F756AA3 44AD8ADD

A15193A9 6CA727E3 A9D8F32E B20E9760 178862DF C154A308
2EAD2E4A 2F7C760A 61E86750 D8B3CFDC 3D00A46A E787A3DE
5F657ADA 64CD0BFF A3B2B228 88299DE3 100F58B7 CD593D82
E5B31614 997FF1CA 6E87790A 3C8B551B 82606A1C BC80ACD2
37683EE2 427BA87C 06FFC4EC 17504261 E99DE627 8F9EF77B
A6E49A82 9C9F27E9 87812CD2 8A15EEAA 36391E67 3042E8A2
9C76C8A4 DABFDB3D 48D351F5 8B870F9F 609AE941 FFA9AA2D
886045F9 5EE9B836 9E1C9545 BC18D60C AE7F56AE 8A24B387
395F0CFC 99E36A65 AFB8C269 522352A0

F.4.2.2 签名密钥与验证密钥

X = 26534136 393884D5 885F924C 188C83AC 860A2560 17B323B4
118E1B1A

Y = D88E9870 78469EB9 B6296190 64BF87C8 1F5BFFD3 88C03AB8
C6C2048E 5B8D087C 8C1D7A5F E2D3394E F567CA43 8AC89B5A
27F26DF2 CCBFDB4A 4351E2BA 42866D82 72DB1BB2 422157D5
3659BB68 FE50F61C 31573684 E93B19C5 F3F832C5 01B6CD9F
3F0E40FA 9198D358 F2BD39F4 DDC4CC5E BD670CB1 677F4F8F
D0867B95 D9AB1A8E 2A8377A9 8189FEF4 0B24150B 2F7EAD42
BBD61A86 C6150431 87614940 435E55DB 55CA0F37 C6F06142
0577EB51 196E0021 431C94DB D746800A F1B4F23E 86F40B57
AB964428 872EE96D A99E3576 0613D279 2FC3E8A8 0CDD1877
117B9ED5 401511ED 43AC10BF 1E1F7450 01AB51CF 842EF938
386947FE BD4CBE9B 1FC3D785 C937D5CE

F.4.2.3 逐消息数据

K = 717548D2 2C7FAF7E BE1BD5DC AD2C2E47 5DA789D5 68C5E081
236214D0

K^{-1} = 787CEDE9 BD149C67 45C83654 E1F9B472 C736D584 ECD4ACE4
4943FD61

M = "abc"的 ASCII 形式 = 61 62 63

F.4.2.4 签名

R = AF246E82 74F24075 343014D5 FC648CE0 09771BD0 48DF1438
EB0D97E7

$R||M$ = AF246E82 74F24075 343014D5 FC648CE0 09771BD0 48DF1438
EB0D97E7
616263

$h(R||M)$ = 15986F5F E1D8BF68 2BD8FE31 25065C4C D9A77CDD 73EE5E85
01D29EF2

S = 370F4FD9 3E761FD9 B97DC37D 4DAD75E2 4EF3A6FD 3D5AC9F1
867A7E33

F.4.2.5 验证

Π' = 18A358C2 2635ABE3 85E18D55 AF94B75B 36EC2FAE 8E87EA24

C0BBC507 A8790BA9 0CD0A93D C92E4736 E5A050BD DA2BF62E
AA042C69 07D39346 D909753C 4C91024F DC12D5AA 5A98FE7E
EE3A90AD 64DB83E9 7C91EAC5 CDC41AA0 D383B3BA 094B0A58
C64BF470 0FC81437 BAD1A15D 3FD61B53 25E2991E D841D159
EEF740DE C7984657 D4B0BE24 9DB622E1 0936BE55 2AE4005A
59DFA542 A688F0DE 3C5D67EB C5A37ED4 C5A3D0F1 CAF1B8FC
24E7E658 368FF8C1 E10A3F80 BE42B30C 6255B2AE 15F0251C
CF1CD9F5 41ECCCF7 7E6A6147 B79FE310 09C8172D BD0CD110
52816FA7 04FADDA3 5C73E38F 9C6BD8BA 131A2073 191D9D64
94A3E764 4ED385FF E01D85BD A015FB6C

R' = AF246E82 74F24075 343014D5 FC648CE0 09771BD0 48DF1438
EB0D97E7

F.5 SDSA 机制

F.5.1 示例 1：2 048 位素数p，SHA-224

F.5.1.1 概述

本示例使用 RFC-5114 中定义的“具有 224 位素数阶子群的 2 048 位 MODP 群”作为群。SHA-224 被专门用作哈希函数，哈希码即为 SHA-224 的输出值，并按照附录*B*转换为相应的数据项。

F.5.1.2 参数

α = 2 048

β = 224

p = AD107E1E 9123A9D0 D660FAA7 9559C51F A20D64E5 683B9FD1
B54B1597 B61D0A75 E6FA141D F95A56DB AF9A3C40 7BA1DF15
EB3D688A 309C180E 1DE6B85A 1274A0A6 6D3F8152 AD6AC212
9037C9ED EFDA4DF8 D91E8FEF 55B7394B 7AD5B7D0 B6C12207
C9F98D11 ED34DBF6 C6BA0B2C 8BBC27BE 6A00E0A0 B9C49708
B3BF8A31 70918836 81286130 BC8985DB 1602E714 415D9330
278273C7 DE31EFDC 7310F712 1FD5A074 15987D9A DC0A486D
CDF93ACC 44328387 315D75E1 98C641A4 80CD86A1 B9E587E8
BE60E69C C928B2B9 C52172E4 13042E9B 23F10B0E 16E79763
C9B53DCF 4BA80A29 E3FB73C1 6B8E75B9 7EF363E2 FFA31F71
CF9DE538 4E71B81C 0AC4DFFE 0C10E64F

G = AC4032EF 4F2D9AE3 9DF30B5C 8FFDAC50 6CDEBE7B 89998CAF
74866A08 CFE4FFE3 A6824A4E 10B9A6F0 DD921F01 A70C4AFA
AB739D77 00C29F52 C57DB17C 620A8652 BE5E9001 A8D66AD7
C1766910 1999024A F4D02727 5AC1348B B8A762D0 521BC98A
E2471504 22EA1ED4 09939D54 DA7460CD B5F6C6B2 50717CBE
F180EB34 118E98D1 19529A45 D6F83456 6E3025E3 16A330EF
BB77A86F 0C1AB15B 051AE3D4 28C8F8AC B70A8137 150B8EEB
10E183ED D19963DD D9E263E4 770589EF 6AA21E7F 5F2FF381
B539CCE3 409D13CD 566AFBB4 8D6C0191 81E1BCFE 94B30269
EDFE72FE 9B6AA4BD 7B5A0F1C 71CFFF4C 19C418E1 F6EC0179

81BC087F 2A7065B3 84B890D3 191F2BFA

q = 801C0D34 C58D93FE 99717710 1F80535A 4738CEBC BF389A99 B36371EB

F.5.1.3 签名密钥与验证密钥

X = 602FE736 80BEFCB2 A8B46779 35FF652B 21A3F4DE 46725D07 D7D371A9

Y = A7DBB446 FD8C4B82 61BE026D 94FA9847 74B17110 CABC0944 14AD2013 2EFC8B7D 7C7FF05D D0B902C4 EF736831 61C1F9A3 9D60E7AB E3BD9FE2 B458A96D F4783408 0AA93CAF 09673967 F434548D 44B278E0 4C1FA6D5 E0C41990 CEF37940 66015ED4 748DAD56 429596DD 9259C45C 21B71A5E A4EF099A 06DAC737 8958A107 B11B3E57 384118E0 19897C48 E734F069 E717E23A DD202405 823A2AE7 A08AFA51 09D2CF6A 0FF546FA 38A8735D 1CE715E0 6AC08EB0 93EB331F EBEC88D6 1DF546E2 DC9E8465 10B63F6A 5BA73FE3 6995BD17 1B9D7D35 9EEB3D7D B801F382 1D582280 DF71A27D 5191E4AB 42BE27A3 1180F537 CABAC0D1 2EBF1698 B1884697 9EDCA15D DC8F86DC

F.5.1.4 逐消息数据

M = "abc"的 ASCII 形式 = 616263

K = 7BFA2DD5 6B31BB27 FFC0D1AE 1ABAA90F A0BB9379 08A542A1 5EFD1E15

Π = 60FCB613 40799851 B5E2DC3A 3865BC21 29100D38 4B1C9A94 6F0C873B 442BEBD8 5904CD09 A4C6A29E 0CD1111E B9E65F82 85F8A578 A5717098 FA2A601F D9183CDD D5FF1586 AB255E1D 4DF4A141 DFE717DC 16DA3B0D 438B1EA5 4976523F 1D73351B F39B1987 97DA0EC7 E9EE994A 4C0352D8 271D186A 0DEA8AB0 FD5E7862 17016E91 03C5F139 2C1D3C01 B974BADC 88184905 065F8DA8 55656BAF B3B1EDBC 4C14A969 2AEA1A71 D85117F4 08548EF5 A34966B9 0123FC81 72472B44 06D0C2E6 77C3C21D D0680C63 0DC69BFF BA67D89F F17CA52A 6B0F164F 5452777D B838BDBD EE60E03B AE773475 42435BD9 09D021DD F97602E0 3FE41463 CC18128B AFA6E661 6F6CA744

R = CC192C12 72872367 48346281 4DF721E5 D9B2A651 0D97F3D3 316AD681

F.5.1.5 签名

R = CC192C12 72872367 48346281 4DF721E5 D9B2A651 0D97F3D3 316AD681

S = 4C776699 9F9D52E1 52B47F29 335E548F A3B90625 C55FC9A4 FE2D5F1F

F.5.1.6 验证

Π' = 60FCB613 40799851 B5E2DC3A 3865BC21 29100D38 4B1C9A94

6F0C873B 442BEBD8 5904CD09 A4C6A29E 0CD1111E B9E65F82
85F8A578 A5717098 FA2A601F D9183CDD D5FF1586 AB255E1D
4DF4A141 DFE717DC 16DA3B0D 438B1EA5 4976523F 1D73351B
F39B1987 97DA0EC7 E9EE994A 4C0352D8 271D186A 0DEA8AB0
FD5E7862 17016E91 03C5F139 2C1D3C01 B974BADC 88184905
065F8DA8 55656BAF B3B1EDBC 4C14A969 2AEA1A71 D85117F4
08548EF5 A34966B9 0123FC81 72472B44 06D0C2E6 77C3C21D
D0680C63 0DC69BFF BA67D89F F17CA52A 6B0F164F 5452777D
B838BDBD EE60E03B AE773475 42435BD9 09D021DD F97602E0
3FE41463 CC18128B AFA6E661 6F6CA744

R' = CC192C12 72872367 48346281 4DF721E5 D9B2A651 0D97F3D3
316AD681

F.5.2 示例 2：2 048 位素数p，SHA-256

F.5.2.1 概述

本示例使用 RFC-5114 中定义的“具有 256 位素数阶子群的 2 048 位 MODP 群”作为群。SHA-256 被专门用作哈希函数，哈希码即为 SHA-256 的输出值，并按照附录*B*转换为相应的数据项。

F.5.2.2 参数

α = 2 048

β = 256

p = 87A8E61D B4B6663C FFBBD19C 65195999 8CEEF608 660DD0F2
5D2CEED4 435E3B00 E00DF8F1 D61957D4 FAF7DF45 61B2AA30
16C3D911 34096FAA 3BF4296D 830E9A7C 209E0C64 97517ABD
5A8A9D30 6BCF67ED 91F9E672 5B4758C0 22E0B1EF 4275BF7B
6C5BFC11 D45F9088 B941F54E B1E59BB8 BC39A0BF 12307F5C
4FDB70C5 81B23F76 B63ACAE1 CAA6B790 2D525267 35488A0E
F13C6D9A 51BFA4AB 3AD83477 96524D8E F6A167B5 A41825D9
67E144E5 14056425 1CCACB83 E6B486F6 B3CA3F79 71506026
C0B857F6 89962856 DED4010A BD0BE621 C3A3960A 54E710C3
75F26375 D7014103 A4B54330 C198AF12 6116D227 6E11715F
693877FA D7EF09CA DB094AE9 1E1A1597

G = 3FB32C9B 73134D0B 2E775066 60EDBD48 4CA7B18F 21EF2054
07F4793A 1A0BA125 10DBC150 77BE463F FF4FED4A AC0BB555
BE3A6C1B 0C6B47B1 BC3773BF 7E8C6F62 901228F8 C28CBB18
A55AE313 41000A65 0196F931 C77A57F2 DDF463E5 E9EC144B
777DE62A AAB8A862 8AC376D2 82D6ED38 64E67982 428EBC83
1D14348F 6F2F9193 B5045AF2 767164E1 DFC967C1 FB3F2E55
A4BD1BFF E83B9C80 D052B985 D182EA0A DB2A3B73 13D3FE14
C8484B1E 052588B9 B7D2BBD2 DF016199 ECD06E15 57CD0915
B3353BBB 64E0EC37 7FD02837 0DF92B52 C7891428 CDC67EB6
184B523D 1DB246C3 2F630784 90F00EF8 D647D148 D4795451
5E2327CF EF98C582 664B4C0F 6CC41659

q = 8CF83642 A709A097 B4479976 40129DA2 99B1A47D 1EB3750B
A308B0FE 64F5FBD3

F.5.2.3 签名密钥与验证密钥

X = 73018895 20D47AA0 55995BA1 D8FCD701 6EA62E09 18892E07
B7DC23AF 69006B88

Y = 57A17258 D4A3F47C 4545AD51 F3109C5D B41B7878 79FCFE53
8DC1DD5D 35CE42FF 3A9F225E DE650212 6408FCB1 3AEA2231
80B149C4 64E176EB F03BA651 0D8206C9 20F6B1E0 9392E6C8
40A05BDB 9D6875AB 3F4817EC 3A65A665 B788ECBB 447188C7
DF2EB4D3 D9424E57 D964398D BE1C6362 659C6BD8 55C1D3E5
1D64796C A598480D FDD9580E 55085345 C15E34D6 A33A2F43
E222407A CE058972 D34952AE 2B705C53 2243BE39 4B222329
6161145E F2927CDBC55BBD56 4AAE8DE4 BA4500A7 FA432FE7
8B0F0689 1E408083 7E761057 BC6CB8AC 18FD4320 7582032A
FB63C624 F32E66B0 5FC31C5D FFB25FA9 2D4D00E2 B0D4F721
E88C417D 2E57797B 8F55A2FF C6EE4DDB

F.5.2.4 逐消息数据

M = "abc"的 ASCII 形式 = 616263

K = 2B73E8FF 3A7C0168 6CA556E0 FABFD74A C8D1FDA4 AD3D503F
23B8EB8A EEC63305

Π = 41979DBA 19606871 A25BBB51 665AD584 9511D125 8C077E93
027D79AC 35EE887C 460C2689 3D7FFC59 0F317B7A 2C01FCB4
3667E373 F8F2D239 0C950BAF 972E6E0B 00A79416 9696CC95
08A895D6 3F8AA7EA 564AA5DE 6104920A 5E9F687F 469BC831
0C02BE30 B8FD4A54 A167E114 01FEE171 EF284811 6146CC69
C0899526 7186C43E 98B5E62E 9CE5C344 646950EF F57B1490
27FE078F CD9C8297 4AFF1DDE 5AF18E4C A3E3787F 66D15D23
292B5F4E 225AAC64 FC904AB3 40A88B76 40F2D436 D2840185
077EACA4 EE75113E 95B26149 F7C6D2CD 554463F9 26E48F09
AD3C99B8 5EA3EC39 E795FEAE C90C8293 FB0D0506 0FE2BF91
5F00E2FB 7C17B2E8 7C462ED0 D49B8E2F

R = CDAC932A 758FCFCE 7E549903 FD891F41 FB5410CB DDD246F3
D6DB0CE6 E0ED696E

F.5.2.5 签名

R = CDAC932A 758FCFCE 7E549903 FD891F41 FB5410CB DDD246F3
D6DB0CE6 E0ED696E

S = 3505AEA2 E039E18F DDC6580A E89E15DF 0103FB45 C1BB763E
DA4EE6F5 F01783CE

F.5.2.6 验证

Π' = 41979DBA 19606871 A25BBB51 665AD584 9511D125 8C077E93
027D79AC 35EE887C 460C2689 3D7FFC59 0F317B7A 2C01FCB4

3667E373 F8F2D239 0C950BAF 972E6E0B 00A79416 9696CC95
08A895D6 3F8AA7EA 564AA5DE 6104920A 5E9F687F 469BC831
0C02BE30 B8FD4A54 A167E114 01FEE171 EF284811 6146CC69
C0899526 7186C43E 98B5E62E 9CE5C344 646950EF F57B1490
27FE078F CD9C8297 4AFF1DDE 5AF18E4C A3E3787F 66D15D23
292B5F4E 225AAC64 FC904AB3 40A88B76 40F2D436 D2840185
077EACA4 EE75113E 95B26149 F7C6D2CD 554463F9 26E48F09
AD3C99B8 5EA3EC39 E795FEAE C90C8293 FB0D0506 0FE2BF91
5F00E2FB 7C17B2E8 7C462ED0 D49B8E2F

R' = CDAC932A 758FCFCE 7E549903 FD891F41 FB5410CB DDD246F3
D6DB0CE6 E0ED696E

F.6 EC-DSA 机制

F.6.1 概述

在以下示例中，SHA-1 被专门用作哈希函数，哈希码即为 SHA-1 的输出值，并按照附录B转换为相应的数据项。

从安全角度来看，应避免使用密码学上弱的曲线（例如，应确保特定曲线不易受到椭圆曲线离散对数问题特殊实例攻击的影响）。

F.6.2 示例 1：域F_{2^m}，$m = 191$，SHA-1

F.6.2.1 参数

域F_{2^m}表示为对不可约多项式$x^{191} + x^9 + 1$取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 2866537B 67675263 6A68F565 54E12640 276B649E F7526267

b = 2E45EF57 1F00786F 67B0081B 9495A3D9 5462F5DE 0AA185EC

G = (G_X, G_Y)

G_X = 36B3DAF8 A23206F9 C4F299D7 B21A9C36 9137F2C8 4AE1AA0D

G_Y = 765BE734 33B3F95E 332932E7 0EA245CA 2418EA0E F98018FB

q = 40000000 00000000 00000000 04A20E90 C39067C8 93BBB9A5

F.6.2.2 签名密钥与验证密钥

X = 340562E1 DDA332F9 D2AEC168 249B5696 EE39D0ED 4D03760F

Y = (Y_X, Y_Y)

Y_X = 5DE37E75 6BD55D72 E3768CB3 96FFEB96 2614dEA4 CE28A2E7

Y_Y = 55C0E0E0 2F5FB132 CAF416EF85B229BBB8E13520 03125BA1

F.6.2.3 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = 3EEACE72 B4919D99 1738D521 879F787C B590AFF8 189D2B69

Π = (Π_X, Π_Y)

Π_X = 438E5A11 FB55E4C6 5471DCD4 9E266142 A3BDF2BF 9D5772D5

Π_Y = 2AD603A0 5BD1D177 649F9167 E6F475B7 E2FF590C 85AF15DA

$h(M)$ = A9993E36 4706816A BA3E2571 7850C26C 9CD0D89D

F.6.2.4 签名

R = 038E5A11 FB55E4C6 5471DCD4 998452B1 E02D8AF7 099BB930

S = 0C9A08C3 4468C244 B4E5D6B2 1B3C6836 28074160 20328B6E

F.6.2.5 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 438E5A11 FB55E4C6 5471DCD4 9E266142 A3BDF2BF 9D5772D5

Π'_Y = 2AD603A0 5BD1D177 649F9167 E6F475b7 E2FF590C 85AF15DA

R' = 038E5A11 FB55E4C6 5471DCD4 998452B1 E02D8AF7 099BB930

F.6.3 示例 2：域F_p，192 位素数p，SHA-1

F.6.3.1 参数

域为F_p，其中p的十六进制表示如下：

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFC

b = 64210519 E59C80E7 0FA7E9AB 72243049 FEB8DEEC C146B9B1

G = (G_X, G_Y)

G_X = 188DA80E B03090F6 7CBF20EB 43A18800 F4FF0AFD 82FF1012

G_Y = 07192B95 FFC8DA78 631011ED 6B24CDD5 73F977A1 1E794811

q = FFFFFFFF FFFFFFFF FFFFFFFF 99DEF836 146BC9B1 B4D22831

F.6.3.2 签名密钥与验证密钥

X = 1A8D598F C15BF0FD 89030B5C B1111AEB 92AE8BAF 5EA475FB

Y = (Y_X, Y_Y)

Y_X = 62B12D60 690CDCF3 30BABAB6 E69763B4 71F994DD 702D16A5

Y_Y = 63BF5EC0 8069705F FFF65E5C A5C0D697 16DFCB34 74373902

F.6.3.3 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

$h(M)$ = A9993E36 4706816A BA3E2571 7850C26C 9CD0D89D

K = FA6DE297 46BBEB7F 8BB1E761 F85F7DFB 2983169D 82FA2F4E

Π = (Π_X, Π_Y)

Π_X = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

Π_Y = 9CF9FA1C BEFEFB91 7747A3BB 29C072B9 289C2547 884FD835

F.6.3.4 签名

R = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

S = E9ECC781 06DEF82B F1070CF1 D4D804C3 CB390046 951DF686

F.6.3.5 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

Π'_Y = 9CF9FA1C BEFEFB91 7747A3BB 29C072B9 289C2547 884FD835

R' = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

F.6.4 示例 3：域F_{2^m}，$m = 283$，SHA-256

F.6.4.1 参数

域F_{2^m}表示为对不可约多项式$x^{283} + x^{12} + x^7 + x^5 + 1$取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 1

b = 027B680A C8B8596D A5A4AF8A 19A0303F CA97FD76 45309FA2
A581485A F6263E31 3B79A2F5

G = (G_X, G_Y)

G_X = 05F93925 8DB7DD90 E1934F8C 70B0DFEC 2EED25B8 557EAC9C
80E2E198 F8CDBECD 86B12053

G_Y = 03676854 FE24141C B98FE6D4 B20D02B4 516FF702 350EDDB0
826779C8 13F0DF45 BE8112F4

q = 03FFFFFF FFFFFFFF FFFFFFFF FFFFEF90 399660FC
938A9016 5B042A7C EFADB307

F.6.4.2 签名密钥与验证密钥

X = 010652D3 7B0A9DB6 4D4033AC 6549CD1D F37E1EED E2612C23
63257C6A FF6C8CB5 DCB63648

Y = (Y_X, Y_Y)

Y_X = 0390858E 9327A714 C74AF0C3 ADEDF4E6 C75CAFDC C46507A4
9E415B13 8A094B6F 43E882AC

Y_Y = 00D4A65D 973CD150 A5221BED F872A4BA 207FF442 7DFFFD48
27C5BF16 9E719162 504D0631

F.6.4.3 逐消息数据

M = "Example of ECDSA with B-283"的 ASCII 形式

$h(M)$ = F0BF4AEF 3F694EBD DE0A7944 5C897ADB 2430B918 77C772DA
9B7362CB 03AEA87F

K = 0100EC32 1393E6DD 6C4D47BE 5AE189E5 E3540857 9D086217
8F94CCBB A3C4049A 4D88E297

$\Pi = (\Pi_X, \Pi_Y)$

Π_X = 077CB284 AC41E72E DA2A93EB 8D6DFF58 620F6C69 D528DFE9
0D909AA5 CABC03A3 4E5D5A76

F.6.4.4 签名

R = 037CB284 AC41E72E DA2A93EB 8D6DFF58 620F7CD9 9B927EEC
7A060A8F 6FB7D926 5EAFA76F

S = 00A37AC1 0AEBFC22 FC6E6EE2 2E8F235E 3EEB0555 A0F0F9DA
92D9FFA7 34AD7679 56D27F23

F.6.4.5 验证

$\Pi' = (\Pi'_X, \Pi'_Y)$

Π'_X = 077CB284 AC41E72E DA2A93EB 8D6DFF58 620F6C69 D528DFE9
0D909AA5 CABC03A3 4E5D5A76

R' = 037CB284 AC41E72E DA2A93EB 8D6DFF58 620F7CD9 9B927EEC
7A060A8F 6FB7D926 5EAFA76F

F.6.5 示例 4：域F_p，256 位素数p，SHA-256

F.6.5.1 参数

域为F_p，其中p的十六进制表示如下：

p = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF
FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF
FFFFFFFC

b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6
3BCE3C3E 27D2604B

$G = (G_X, G_Y)$

G_X = 6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0
F4A13945 D898C296

G_Y = 4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE
CBB64068 37BF51F5

q = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84
F3B9CAC2 FC632551

F.6.5.2 签名密钥与验证密钥

X = C477F9F6 5C22CCE2 0657FAA5 B2D1D812 2336F851 A508A1ED
04E479C3 4985BF96

$Y = (Y_X, Y_Y)$

Y_X = B7E08AFD FE94BAD3 F1DC8C73 4798BA1C 62B3A0AD 1E9EA2A3
8201CD08 89BC7A19

Y_Y = 3603F747 959DBF7A 4BB226E4 19287290 63ADC7AE 43529E61
B563BBC6 06CC5E09

F.6.5.3 逐消息数据

M = "Example of ECDSA with P-256"的 ASCII 形式

$h(M)$ = A41A41A1 2A799548 211C410C 65D8133A FDE34D28 BDD542E4
B680CF28 99C8A8C4

K = 7A1A7E52 797FC8CA AA435D2A 4DACE391 58504BF2 04FBE19F
14DBB427 FAEE50AE

Π = (Π_X, Π_Y)

Π_X = 2B42F576 D07F4165 FF65D1F3 B1500F81 E44C316F 1F0B3EF5
7325B69A CA46104F

F.6.5.4 签名

R = 2B42F576 D07F4165 FF65D1F3 B1500F81 E44C316F 1F0B3EF5
7325B69A CA46104F

S = DC42C212 2D6392CD 3E3A993A 89502A81 98C1886F E69D262C
4B329BDB 6B63FAF1

F.6.5.5 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 2B42F576 D07F4165 FF65D1F3 B1500F81 E44C316F 1F0B3EF5
7325B69A CA46104F

R' = 2B42F576 D07F4165 FF65D1F3 B1500F81 E44C316F 1F0B3EF5
7325B69A CA46104F

F.6.6 示例 5：域F_p，192 位素数p，SHA-224

F.6.6.1 参数

域为F_p，其中p的十六进制表示如下：

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFC

b = 64210519 E59C80E7 0FA7E9AB 72243049 FEB8DEEC C146B9B1

G = (G_X, G_Y)

G_X = 188DA80E B03090F6 7CBF20EB 43A18800 F4FF0AFD 82FF1012

G_Y = 07192B95 FFC8DA78 631011ED 6B24CDD5 73F977A1 1E794811

q = FFFFFFFF FFFFFFFF FFFFFFFF 99DEF836 146BC9B1 B4D22831

F.6.6.2 签名密钥与验证密钥

X = 1A8D598F C15BF0FD 89030B5C B1111AEB 92AE8BAF 5EA475FB

Y = (Y_X, Y_Y)

Y_X = 62B12D60 690CDCF3 30BABAB6 E69763B4 71F994DD 702D16A5

Y_Y = 63BF5EC0 8069705F FFF65E5C A5C0D697 16DFCB34 74373902

F.6.6.3 逐消息数据

M = “abc”的 ASCII 形式 = 61 62 63

$h(M)$ = 23097D22 3405D822 8642A477 BDA255B3 2AADBCE4 BDA0B3F7 E36C9DA7

H = 23097D22 3405D822 8642A477 BDA255B3 2AADBCE4 BDA0B3F7

K = FA6DE297 46BBEB7F 8BB1E761 F85F7DFB 2983169D 82FA2F4E

Π = (Π_X, Π_Y)

Π_X = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

Π_Y = 9CF9FA1C BEFEFB91 7747A3BB 29C072B9 289C2547 884FD835

F.6.6.4 签名

R = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

S = 6663F278 36987EED 458582D1 F443293A 5ED88849 B2FC5CD9

F.6.6.5 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

Π'_Y = 9CF9FA1C BEFEFB91 7747A3BB 29C072B9 289C2547 884FD835

R' = 88505238 0FF147B7 34C330C4 3D39B2C4 A89F29B0 F749FEAD

F.6.7 示例 6：域F_{2^m}，$m = 233$，SHA-256

F.6.7.1 参数

域F_{2^m}表示为对不可约多项式$x^{233} + x^{74} + 1$ 取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 0

b = 1

G = (G_X, G_Y)

G_X = 0172 32BA853A 7E731AF1 29F22FF4 149563A4 19C26BF5 0A4C9D6E EFAD6126

G_Y = 01DB 537DECE8 19B7F70F 555A67C4 27A8CD9B F18AEB9B 56E0C110 56FAE6A3

q = 0080 00000000 00000000 00000000 00069D5B B915BCD4 6EFB1AD5 F173ABDF

F.6.7.2 签名密钥与验证密钥

X = 8434613F 4B799B4C 26E4D7AB 8E9481B0 4B09E648 C94AFFD1 4B611A20

Y = (Y_X, Y_Y)

Y_X = 017C 9DD766AE FBE4DE4B 15F46DB0 671DC4CA 0767ED51 ECEA9475 7D9C662E

Y_Y = 01CD D7260848 37AE73C1 1C27D605 C6EB2D5E 31482358 780305C2 522B151B

F.6.7.3 逐消息数据

M = “abc”的 ASCII 形式 = 61 62 63

$h(M)$ = BA7816BF 8F01CFEA 414140DE 5DAE2223 B00361A3 96177A9C B410FF61 F20015AD

H = BA 7816BF8F 01CFEA41 4140DE5D AE2223B0 0361A396 177A9CB4 10FF61F2

K = 0001 90DA60FE 3B179B96 611DB7C7 E5217C9A FF0AEE43 5782EBFB 2DFFF27F

Π = (Π_X, Π_Y)

Π_X = 01BE A7231662 E6516F11 E37D59D5 00EAE71D 116E9B7B BCE5964B 88D4CC4D

Π_Y = 00C9 8F8C9A7D 65880920 C2FEBE55 2D824597 9E6D67CE 82A41EF1 BAD22FD3

F.6.7.4 签名

R = 003E A7231662 E6516F11 E37D59D5 00D70F09 E62D64FE 6FF445C9 B479C8B0

S = 002D 72C73DA3 3A9B267F 0BEC9E6C B6BECEED 014F67D4 A3D30006 B3EBE2DC

F.6.7.5 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 01BE A7231662 E6516F11 E37D59D5 00EAE71D 116E9B7B BCE5964B 88D4CC4D

Π'_Y = 00C9 8F8C9A7D 65880920 C2FEBE55 2D824597 9E6D67CE 82A41EF1 BAD22FD3

R' = 003E A7231662 E6516F11 E37D59D5 00D70F09 E62D64FE 6FF445C9 B479C8B0

F.6.8 示例 7：域F_{2^m}，$m = 283$，SHA-384

F.6.8.1 参数

域F_{2^m}表示为对不可约多项式$x^{283} + x^{12} + x^7 + x^5 + 1$取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 0

b = 1

G = (G_X, G_Y)

G_X = 0503213F 78CA4488 3F1A3B81 62F188E5 53CD265F 23C1567A
16876913 B0C2AC24 58492836

G_Y = 01CCDA38 0F1C9E31 8D90F95D 07E5426F E87E45C0 E8184698
E4596236 4E341161 77DD2259

q = 01FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFE9AE 2ED07577
265DFF7F 94451E061E163C61

F.6.8.2 签名密钥与验证密钥

X = 00069E6D 19F7E454 A83664FF 49208F60 38EAF842 E164DF42
D0F64948 FF9C94B0 14988329

Y = (Y_X, Y_Y)

Y_X = 01B64A60 D4A36540 9635AAA2 7E1708D9 0B839AFA 2D9820E1
2B79C3AF 1094B601 0AAEF5BE

Y_Y = 0334B5F3 0CA21756 BDE6D477 38F2458F 56FBF6BD C76FCFB8
F3E59145 5F041A95 2EE87A8E

F.6.8.3 逐消息数据

M = “abc”的 ASCII 形式 = 61 62 63

$h(M)$ = CB00753F 45A35E8B B5A03D69 9AC65007 272C32AB 0EDED163
1A8B605A 43FF5BED 8086072B A1E7CC23 58BAECA1 34C825A7

H = 019600EA 7E8B46BD 176B407A D3358CA0 0E4E5865 561DBDA2
C63516C0 B487FEB7 DB010C0E

K = E308 4442D66F A9A02C42 890163E5 7EE33CA1 F4583C65
BCBDE927 81C7A3C8 3E89B773

Π = (Π_X, Π_Y)

Π_X = 07C973D5 8FD17A06 AA8F39D5 EC42E0A6 B992F6CC 61F15756
5DD7036C 147D9005 400C1328

Π_Y = 012EB10A BED281AE DDA27842 3ECB4514 5E59AEFB 5838C287
AFD981F0 D90238E0 A8B13720

F.6.8.4 签名

R = 01C973D5 8FD17A06 AA8F39D5 EC42E0A6 B99339C1 D57FF6F0
EABD04ED 57AE35F2 E5C95E05

S = 01634F16 C591CFC8 19DD0F33 72261EE6 0596515B 85CAF647
A0369888 B9A1866D 865CF19D

F.6.8.5 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 07C973D5 8FD17A06 AA8F39D5 EC42E0A6 B992F6CC 61F15756
5DD7036C 147D9005 400C1328

Π'_Y = 012EB10A BED281AE DDA27842 3ECB4514 5E59AEFB 5838C287
AFD981F0 D90238E0 A8B13720

R' = 01C973D5 8FD17A06 AA8F39D5 EC42E0A6 B99339C1 D57FF6F0
EABD04ED 57AE35F2 E5C95E05

F.7 EC-KCDSA 机制

F.7.1 示例 1：域F_p，224 位素数p，SHA-224

F.7.1.1 概述

本示例使用 SHA-224 作为哈希函数h。哈希码即为 SHA-224 的输出值。

F.7.1.2 参数

域为F_p，其中p为

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 00000000
00000001

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF
FFFFFFFE

b = B4050A85 0C04B3AB F5413256 5044B0B7 D7BFD8BA 270B3943
2355FFB4

G = (G_X, G_Y)

G_X = B70E0CBD 6BB4BF7F 321390B9 4A03C1D3 56C21122 343280D6
115C1D21

G_Y = BD376388 B5F723FB 4C22DFE6 CD4375A0 5A074764 44D58199
85007E34

q = FFFFFFFF FFFFFFFF FFFFFFFF FFFF16A2 E0B8F03E 13DD2945
5C5C2A3D

F.7.1.3 签名密钥与验证密钥

X = 562A6F64 E162FFCB 51CD4707 774AE366 81B6CEF2 05FE5D43
912956A2

Y = (Y_X, Y_Y)

Y_X = B574169E 4FCEF1AF 3429D8BB 5481FF7D FA978690 492E1098
B80A5579

Y_Y = 1576819B D9F0B685 19EE844A FE88CCFB 2AD574A5 6472D954
1461AE7E

F.7.1.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76

61 6C 69 64 61 74 69 6F 6E 2E

K = 76A0AFC1 8646D1B6 20A079FB 223865A7 BCB447F3 C03A35D8 78EA4CDA

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = F887C158 65203FF7 2C69C113 A457DF64 4F627801 DFF99D1B CCB49C2D

Π_Y = ADE18B5B B7118745 017631E5 E54B36C0 332D70B3 CAA8FB10 728B66E0

Y' = B574169E 4FCEF1AF 3429D8BB 5481FF7D FA978690 492E1098 B80A5579 1576819B D9F0B685 19EE844A FE88CCFB 2AD574A5 6472D954 1461AE7E 00000000 00000000

$h(Y'\|M)$ = 8C5CB967 71166477 FF84D281 DB766201 2F842138 8AA6FC05 282E2E03

F.7.1.5 签名

R = EEA58C91 E0CDCEB5 799B00D2 412D928F DD23122A 1C2BDF43 C2F8DAFA

S = AEBAB53C 7A44A8B2 2F35FDB9 DE265F23 B89F65A6 9A8B7BD4 061911A6

F.7.1.6 验证

R' = EEA58C91 E0CDCEB5 799B00D2 412D928F DD23122A 1C2BDF43 C2F8DAFA

F.7.2 示例 2：域F_p，256 位素数p，SHA-256

F.7.2.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。

F.7.2.2 参数

域为F_p，其中p为

p = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFC

b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6 3BCE3C3E 27D2604B

G = (G_X, G_Y)

G_X = 6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0 F4A13945 D898C296

G_Y = 4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE CBB64068 37BF51F5

q = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84 F3B9CAC2 FC632551

F.7.2.3 签名密钥与验证密钥

X = 9051A275 AA4D9843 9EDDED13 FA1C6CBB CCE775D8 CC9433DE
E69C5984 8B3594DF

Y = (Y_X, Y_Y)

Y_X = 148EDDD3 734FD5F1 5987579F 516089A8 C9FEF4AB 76B59D7B
8A01CDC5 6C4EDFDF

Y_Y = A4E2E42C B4372A6F 2F3F71A1 49481549 F68D2963 539C853E
46B94696 569E8D61

F.7.2.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76
61 6C 69 64 61 74 69 6F 6E 2E

K = 71B88F39 8916DA9C 90F555F1 B5732B7D C636B49C 638150BA
C11BF05C FE16596A

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = EC3847B0 CA52038A 823D0230 14546B41 4946EF0A 6EE09228
38948459 5F30E26C

Π_Y = 0640451D 36932442 4ABC681D 65653986 6AD9C494 D26FAC14
69FC2A08 D945F130

Y' = 148EDDD3 734FD5F1 5987579F 516089A8 C9FEF4AB 76B59D7B
8A01CDC5 6C4EDFDF A4E2E42C B4372A6F 2F3F71A1 49481549
F68D2963 539C853E 46B94696 569E8D61

$h(Y'\|M)$ = 681C8ED8 9E8B0E1B C369AA10 6F6B9813 E6338F0C 54BE577A
87623492 52F9BEDF

F.7.2.5 签名

R = 0EDDF680 601266EE 1DA83E55 A6D9445F C781DAEB 14C765E7
E5D0CDBA F1F14A68

S = 9B333457 661C7CF7 41BDDBC0 835553DF BB37EE74 F53DB699
E0A17780 C7B6F1D0

F.7.2.6 验证

R' = 0EDDF680 601266EE 1DA83E55 A6D9445F C781DAEB 14C765E7
E5D0CDBA F1F14A68

F.7.3 示例 3：域F_{2^m}，$m = 233$，SHA-224

F.7.3.1 概述

本示例使用 SHA-224 作为哈希函数h。哈希码即为 SHA-224 的输出值。

F.7.3.2 参数

域F_{2^m}表示为对不可约多项式$x^{233} + x^{74} + 1$ 取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 1

b = 0066 647EDE6C 332C7F8C 0923BB58 213B333B 20E9CE42
81FE115F 7D8F90AD

G = (G_X, G_Y)

G_X = 00FA C9DFCBAC 8313BB21 39F1BB75 5FEF65BC 391F8B36
F8F8EB73 71FD558B

G_Y = 0100 6A08A419 03350678 E58528BE BF8A0BEF F867A7CA
36716F7E 01F81052

q = 0100 00000000 00000000 00000000 0013E974 E72F8A69
22031D26 03CFE0D7

F.7.3.3 签名密钥与验证密钥

X = 00BF 83825505 3DBF499C BE190DE3 5BC14AFC 1EA142F3
5EE69838 5B48D688

Y = (Y_X, Y_Y)

Y_X = 01F4 85A65E59 B336E140 1C8A311F 01C92626 C663E69F
12A627E5 3E8F0675

Y_Y = 01BF 338CE75A DFB07DEB D962E1D8 0C101587 269AC995
1B40422B 12E9DA3E

F.7.3.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76
61 6C 69 64 61 74 69 6F 6E 2E

K = 00F4 F088192E 8EB1CD8B 4ECB3A53 33746B40 EBF16966
A213B18A 176B2F62

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = 00E4 5041E7AA 060B8B1A 02A7ACAC DB4E95EF F61F33C0
BB8D6EC2 F1C68BA1

Π_Y = 0155 B3A1DA61 F81E04D5 80D07E92 93DF3D4C 7FE34686
BD157374 4D8D3F18

Y' = 01F485A6 5E59B336 E1401C8A 311F01C9 2626C663 E69F12A6
27E53E8F 067501BF 338CE75A DFB07DEB D962E1D8 0C101587
269AC995 1B40422B 12E9DA3E 00000000

$h(Y'\|M)$ = E74B3C74 72F2E97E C31861CA 1773472E 58828A98 026277CB
00EF36AC

F.7.3.5 签名

R = 82EF9427 4AC70A3D AC231E38 AE0F0D31 8FD8E189 EE40A3E0
61EC80BF

S = 00A8 CD7F7573 BAC3C4C4 00F65FDC CCD46F58 EBFC54CE
45571075 FD7704DB

F.7.3.6 验证

R' = 82EF9427 4AC70A3D AC231E38 AE0F0D31 8FD8E189 EE40A3E0
61EC80BF

F.7.4 示例 4：域F_{2^m}，$m = 233$（Koblitz 曲线），SHA-224

F.7.4.1 概述

本示例使用 SHA-224 作为哈希函数h。哈希码即为 SHA-224 的输出值。本示例使用 Koblitz 曲线作为椭圆曲线。

F.7.4.2 参数

域F_{2^m}表示为对不可约多项式$x^{233} + x^{74} + 1$ 取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 0

b = 1

G = (G_X, G_Y)

G_X = 0172 32BA853A 7E731AF1 29F22FF4 149563A4 19C26BF5
0A4C9D6E EFAD6126

G_Y = 01DB 537DECE8 19B7F70F 56E0C110 56FAE6A3 555A67C4
27A8CD9B F18AEB9B

q = 80 00000000 00000000 00000000 00069D5B B915BCD4
6EFB1AD5 F173ABDF

F.7.4.3 签名密钥与验证密钥

X = 0073 6439374F 72B1C723 AE611CB3 DFBCA0A8 E2C5096B
DB9C2D37 21167B49

Y = (Y_X, Y_Y)

Y_X = 01E9 1DEFBD41 AE655105 E046E03E C13E3860 0E9A2C9
920B8E75 53721605

Y_Y = 0112 9C2706D1 9D134891 C7BAD84A 5600C2AF F86068C4
7497F5BD 498D0B76

F.7.4.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76
61 6C 69 64 61 74 69 6F 6E 2E

K = 0061 7AA0B7A8 197A2B81 01500BFE 55D5322A 7149E275
F91ADBC7 E30128E4

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = 01BB 9CDB150A 2E5669ED C491320C 3F84E28A 7D6631BC
51127677 A2CF2FEF

Π_Y = 00DA E917793C 12DE86AA 6727C396 A3131B69 33344EDD
B621DD29 BC09B648

Y' = 01E91DEF BD41AE65 5105E046 E03EC13E 38600E9A 2C9A920B
8E755372 16050112 9C2706D1 9D134891 C7BAD84A 5600C2AF
F86068C4 7497F5BD 498D0B76 00000000

$h(Y'\|M)$ = FC712972 727661DE B546E86A B6937DB7 D9E61A36 DF5CEA86
044BFF25

F.7.4.5 签名

R = B164A12F 615CC661 C10B78CB 6E01C9DE 46337C50 C036FAC5
51178752

S = 004A 2109081E B3ADF95C 19FFAE89 5D303B83 147B27C6
EFAE8536 2BFAB89A

F.7.4.6 验证

R' = B164A12F 615CC661 C10B78CB 6E01C9DE 46337C50 C036FAC5
51178752

F.7.5 示例 5：域F_{2^m}，$m = 283$，SHA-256

F.7.5.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。

F.7.5.2 参数

域F_{2^m}表示为对不可约多项式$x^{283} + x^{12} + x^7 + x^5 + 1$取模的多项式。

椭圆曲线定·义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 1

b = 027B680A C8B8596D A5A4AF8A 19A0303F CA97FD76 45309FA2
A581485A F6263E31 3B79A2F5

G = (G_X, G_Y)

G_X = 05F93925 8DB7DD90 E1934F8C 70B0DFEC 2EED25B8 557EAC9C
80E2E198 F8CDBECD 86B12053

G_Y = 03676854 FE24141C B98FE6D4 B20D02B4 516FF702 350EDDB0
826779C8 13F0DF45 BE8112F4

q = 03FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFEF90 399660FC
938A9016 5B042A7C EFADB307

F.7.5.3 签名密钥与验证密钥

X = 00D64BEC 51F1ADA0 5BBD4F2B 53405B0C E8A1B99C D8DB6309
76A47F76 F08F205E EFC3FBD8

Y = (Y_X, Y_Y)

Y_X = 04313C7E 9C4F80D2 6A287B37 FE7FAA96 BE31F116 2E18BDB4
70CF43D4 DB28DE10 8B007E9F

Y_Y = 0342CCF6 F502F9DF EC208170 24326C26 E867E1FB EC6634CB
17023CA0 222D6112 E0BFA106

F.7.5.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76
61 6C 69 64 61 74 69 6F 6E 2E

K = 00D18E44 CB7F75F8 01277FA5 CF31A268 8CC2F322 2FA9F26E
E8598126 AFEEE4E3 8DD0E08E

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = 01EBE1E7 8BAF9EF6 833189B3 ACC5B3DC 85788292 E0006D90
F8A4E2AE C3027F28 BE47FACA

Π_Y = 0721C5B3 4A038EE1 1DEE2FAE DA84FD46 CBCF37BA 676677BB
AB731AE8 8C52833B AB776F45

Y' = 04313C7E 9C4F80D2 6A287B37 FE7FAA96 BE31F116 2E18BDB4
70CF43D4 DB28DE10 8B007E9F 0342CCF6 F502F9DF EC208170
24326C26 E867E1FB EC6634CB 17023CA0

$h(Y'\|M)$ = 148DF2CD 1A4E5437 69F5F0B4 FE07A87A D630C512 A3978248
5B8B8A1A EA50D662

F.7.5.5 签名

R = 4A23BA73 B29A9010 ACD1E231 3B9A252C E209C7BF 3643926F
A7BF8C87 A8C76D40

S = 03AA4FFF F1F4C3EE BF9C8798 2E717572 71CB7662 BA03463B
8B5F97B0 5C7F7C2C 88A31799

F.7.5.6 验证

R' = 4A23BA73 B29A9010 ACD1E231 3B9A252C E209C7BF 3643926F
A7BF8C87 A8C76D40

F.7.6 示例 6：域F_{2^m}，$m = 283$（Koblitz 曲线），SHA-256

F.7.6.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。本示例使用 Koblitz 曲线作为椭圆曲线。

F.7.6.2 参数

域F_{2^m}表示为对不可约多项式$x^{283} + x^{12} + x^7 + x^5 + 1$取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 0

b = 1

G = (G_X, G_Y)

G_X = 0503213F 78CA4488 3F1A3B81 62F188E5 53CD265F 23C1567A
16876913 B0C2AC24 58492836

G_Y = 01CCDA38 0F1C9E31 8D90F95D 07E5426F E87E45C0 E8184698
E4596236 4E341161 77DD2259

q = 01FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFE9AE 2ED07577
265DFF7F 94451E06 1E163C61

F.7.6.3 签名密钥与验证密钥

X = 014930E6 6B51F09F EEBBAFFC 9111C5CF 8AE406C9 35AC9618
F0A613B9 6D97F7DB 8F6EBA74

Y = (Y_X, Y_Y)

Y_X = 078A6ACD D5F779F2 5E8AB413 965E217F E6B1E63D 4717EEF5
0DC8C59D F7B1A095 BC3027AE

Y_Y = 07B6D962 5F2D9DDF 516B5037 E1E7B115 26E12AC4 E65AD498
CD85D65A 9E915D58 6976C00F

F.7.6.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式 =

54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76
61 6C 69 64 61 74 69 6F 6E 2E

K = 01EA8FB5 72B7B2DA 7149DCD8 78101ECF 3F296400 E13A0D65
C8B6E558 C0237C6D A55268A1

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = 0227BDFF E74468EE 3A327AAC 7078252F F545113A 1DD9A2E0
7A0D238B AE601410 34D91C33

Π_Y = 02F519CC E08F4ACC 46AB4323 45DD0F69 408E1346 5E017832
94DE4128 4E24D02B D6916937

Y' = 078A6ACD D5F779F2 5E8AB413 965E217F E6B1E63D 4717EEF5
0DC8C59D F7B1A095 BC3027AE 07B6D962 5F2D9DDF 516B5037
E1E7B115 26E12AC4 E65AD498 CD85D65A

$h(Y'\|M)$ = 23893E3F 87BA26BB B05E0E9B F83A40B0 14EFB95B C87B3AF4
C34902D6 12C8A2B4

F.7.6.5 签名

R = E214F3CF 8BBB6E92 F779E6C8 A3424BA8 64734002 5EB49EED
C6016746 81B14AFD

S = 0014CC0B B9245B7A 8BC3C6E0 392AAACE DCED8A61 9D9676E9
73D5244D 7F45E01D B425A93E

F.7.6.6 验证

R' = E214F3CF 8BBB6E92 F779E6C8 A3424BA8 64734002 5EB49EED
C6016746 81B14AFD

F.7.7 示例 7：域F_p，224 位素数p，SHA-256

F.7.7.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。

F.7.7.2 参数

域为F_p，其中p为

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 00000000 00000001

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFE

b = B4050A85 0C04B3AB F5413256 5044B0B7 D7BFD8BA 270B3943 2355FFB4

G = (G_X, G_Y)

G_X = B70E0CBD 6BB4BF7F 321390B9 4A03C1D3 56C21122 343280D6 115C1D21

G_Y = BD376388 B5F723FB 4C22DFE6 CD4375A0 5A074764 44D58199 85007E34

q = FFFFFFFF FFFFFFFF FFFFFFFF FFFF16A2 E0B8F03E 13DD2945 5C5C2A3D

F.7.7.3 签名密钥与验证密钥

X = 61585827 449DBC0E C161B2CF 8575C9DF 149F41DD 0289BE4F F110773D

Y = (Y_X, Y_Y)

Y_X = CFA3A0C1 F4A0903E 84F314B0 70FB3EFA 531DB6D3 86739E01 2609557C

Y_Y = FDB53330 B727A7B3 D40E332B 59AF060C 957D908D 18862159 F92B26B3

F.7.7.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式

= 54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D 65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53 41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76 61 6C 69 64 61 74 69 6F 6E 2E

K = EEC79D8D 4648DF3A 832A66E3 775537E0 00CC9B95 7E1319C5 DB9DD4F7

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = 71A141D9 23FBE6A3 B15130BE FE4400E9 630B8137 9DF0162E AFF6470C

Π_Y = 08CDBF65 B9FF5604 79C19237 D6F1A760 0331F60A 011FC507 79DC4CDC

Y' = CFA3A0C1 F4A0903E 84F314B0 70FB3EFA 531DB6D3 86739E01
2609557C FDB53330 B727A7B3 D40E332B 59AF060C 957D908D
18862159 F92B26B3 00000000 00000000

$h(Y'||M) \bmod 2^{\beta'}$ = 54F85A9A 6E91CEE3 176F05C9 61689D8E 3BDFC29A
54D0E4B0 B842C41F

F.7.7.5 签名

R = 64B49E97 7E6534F8 77CB68A3 806F6A98 9311CEAA 8A64A055
8077C04B

S = AFF23D40 B1779511 51BE32F6 561B1B73 9E3E8F82 2CC52D4C
B3909A93

F.7.7.6 验证

R' = 64B49E97 7E6534F8 77CB68A3 806F6A98 9311CEAA 8A64A055
8077C04B

F.7.8 示例 8：域F_{2^m}，$m = 233$，SHA-256

F.7.8.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。

F.7.8.2 参数

域F_{2^m}表示为对不可约多项式$x^{233} + x^{74} + 1$ 取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 1

b = 0066 647EDE6C 332C7F8C 0923BB58 213B333B 20E9CE42
81FE115F 7D8F90AD

G = (G_X, G_Y)

G_X = 00FA C9DFCBAC 8313BB21 39F1BB75 5FEF65BC 391F8B36
F8F8EB73 71FD558B

G_Y = 0100 6A08A419 03350678 E58528BE BF8A0BEF F867A7CA
36716F7E 01F81052

q = 0100 00000000 00000000 00000000 0013E974 E72F8A69
22031D26 03CFE0D7

F.7.8.3 签名密钥与验证密钥

X = 000E D21C5C28 5F2B454A 0FE5D97C 6A86AA3F 7CB14FFD
D35EB089 BE11F031

Y = (Y_X, Y_Y)

Y_X = 0068 A50FAF91 43203612 1C5B6D2C 9307EA20 1FFE6F74
E09EF223 0AEC930F

Y_Y = 0052 67430AF3 EF4FB190 A4430F26 9521D7FC E007E221
245F5D14 C7541963

F.7.8.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式

= 54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76
61 6C 69 64 61 74 69 6F 6E 2E

K = 0000 A516B3AD 24EB7F85 4D101DDF AB5A09A9 9D2C566A
09B29E57 2DAFDE75

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = 0153 03B54973 AF9A034D 203C50BB 1C98C047 AA01E280
B1CA1214 B5F19A5B

Π_Y = 01E8 6F2226BF 8FD9E15E 3F26B4EF CBE76148 E39C9236
76D3BC09 96634015

Y' = 0068A50F AF914320 36121C5B 6D2C9307 EA201FFE 6F74E09E
F2230AEC 930F0052 67430AF3 EF4FB190 A4430F26 9521D7FC
E007E221 245F5D14 C7541963 00000000

$h(Y'\|M) \bmod 2^{\beta'}$ = F33B 7E265137 98DEA674 DE5697D5 3DCE457B D19C406D
C23C034A A8849855

F.7.8.5 签名

R = E1C9 75FBD0E8 98FDB018 61C4EC8D 4CEAE19B 8CFCBBC8
09EF3A03 AD3A853A

S = 00EF 7A6CAA2B 46D5BE07 DB837779 49F2505C 877FC475
76A54D40 BCD53D5F

F.7.8.6 验证

R' = E1C9 75FBD0E8 98FDB018 61C4EC8D 4CEAE19B 8CFCBBC8
09EF3A03 AD3A853A

F.7.9 示例 9：域F_{2^m}，$m = 233$（Koblitz 曲线），SHA-256

F.7.9.1 概述

本示例使用 SHA-256 作为哈希函数h。哈希码即为 SHA-256 的输出值。本示例使用 Koblitz 曲线作为椭圆曲线。

F.7.9.2 参数

域F_{2^m}表示为对不可约多项式$x^{233} + x^{74} + 1$ 取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 0

b = 1

G = (G_X, G_Y)

G_X = 0172 32BA853A 7E731AF1 29F22FF4 149563A4 19C26BF5
0A4C9D6E EFAD6126

G_Y = 01DB 537DECE8 19B7F70F 555A67C4 27A8CD9B F18AEB9B
56E0C110 56FAE6A3

q = 80 00000000 00000000 00000000 00069D5B B915BCD4
6EFB1AD5 F173ABDF

F.7.9.3 签名密钥与验证密钥

X = 0028 13E18571 44BE8611 A7B93256 4EB3603D B08406A6
90D8185D EFE6EC5F

Y = (Y_X, Y_Y)

Y_X = 00CF 66347977 26F04185 BC953B3D DB4B9375 9D074522
0938DA29 DD7FA585

Y_Y = 01E0 00F206CE D0896589 2BCFA3E7 B459CED4 7188EBDA
7A74B03E 2ECB66FC

F.7.9.4 逐消息数据

M = “This is a sample message for EC-KCDSA implementation validation.”的 ASCII 形式

= 54 68 69 73 20 69 73 20 61 20 73 61 6D 70 6C 65 20 6D
65 73 73 61 67 65 20 66 6F 72 20 45 43 2D 4B 43 44 53
41 20 69 6D 70 6C 65 6D 65 6E 74 61 74 69 6F 6E 20 76
61 6C 69 64 61 74 69 6F 6E 2E

K = 0006 B64D5DB8 65FD5E32 72ABFDBA EF0964F1 C26546A0
1582A4AD E5C0C2CF

Π = $[K]G = (\Pi_X, \Pi_Y)$

Π_X = 003D 7E8D89DF 5276B2C0 4A827557 F24D4F3F B6637CE4
5A826CB5 4B7C236C

Π_Y = 00B0 EF449FF2 8EE97721 4999B77F 7B93AADF E01C4685
20474883 85A9EC70

Y' = 00CF6634 797726F0 4185BC95 3B3DDB4B 93759D07 45220938
DA29DD7F A58501E0 00F206CE D0896589 2BCFA3E7 B459CED4
7188EBDA 7A74B03E 2ECB66FC 00000000

$h(Y'\|M) \bmod 2^{\beta'}$ = 19 2B5AD22F E1C413D1 414499CC 4EBEFD60 7F952216
9ADA39AC 26ABA55E

F.7.9.5 签名

R = D4 B2C6E695 9906C6A6 A8290AEF 7261FE96 EADCC177
63A1DE9D D009737C

S = 08 657B1F0C DFCDF279 A6433A8D 68BA2D02 244A4C34
8519A05F A2CF37ED

F.7.9.6 验证

R' = D4 B2C6E695 9906C6A6 A8290AEF 7261FE96 EADCC177
63A1DE9D D009737C

F.8 EC-GDSA 机制

F.8.1 概述

以下示例使用 Brainpool 曲线和 SHA-2。

F.8.2 示例 1：域F_p，192 位素数p，SHA-256

F.8.2.1 参数

域为F_p，其中p的十六进制表示如下：

p = C302F41D 932A36CD A7A34630 93D18DB7 8FCE476D E1A86297

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = 6A911740 76B1E0E1 9C39C031 FE8685C1 CAE040E5 C69A28EF

b = 469A28EF 7C28CCA3 DC721D04 4F4496BC CA7EF414 6FBF25C9

G = (G_X, G_Y)

G_X = C0A0647E AAB6A487 53B033C5 6CB0F090 0A2F5C48 53375FD6

G_Y = 14B69086 6ABD5BB8 8B5F4828 C1490002 E6773FA2 FA299B8F

q = C302F41D 932A36CD A7A3462F 9E9E916B 5BE8F102 9AC4ACC1

F.8.2.2 签名密钥与验证密钥

X = 40F95B49 A3B1BF55 311A56DF D3B5061E E1DF6439 84D41E35

Y = (Y_X, Y_Y)

Y_X = 754A8F6C 30D28AE9 A63443C9 7DEC844A 15F797D0 B78FEE03

Y_Y = 63EC81B4 6A9F3833 025037DF E7DCDDE7 AF20C5E7 C6733C35

F.8.2.3 逐消息数据

M = "brainpoolP192r1"的 ASCII 形式 =

62 72 61 69 6E 70 6F 6F 6C 50 31 39 32 72 31

K = 5A966260 96288CC4 69F1704E C05F44D1 EC18BD32 CEB02D5B

Π = (Π_X, Π_Y)

Π_X = A00B0AA2 5DB6AB5C 21B86300 D9BC99F5 6E9DD1B7 F1DC4774

Π_Y = 58C0F50E 2E1F6B01 A50E280E 6DB71637 AE9579BC 1565F369

$h(M)$ = 2AE5880D 61FCA83B 2D4C9281 356B9FD2 F7C21359 BA789FBF
D7068AF2 F9A101EC

H = 2AE5880D 61FCA83B 2D4C9281 356B9FD2 F7C21359 BA789FBF

（H为消息M的 SHA-256 哈希码截断至与q相同的比特长度后的结果。）

F.8.2.4 签名

R = A00B0AA2 5DB6AB5C 21B86300 D9BC99F5 6E9DD1B7 F1DC4774

S = 634635EF 813247D7 20245C94 09FB20A2 67C560C8 8EB2B07B

F.8.2.5 验证

R' = A00B0AA2 5DB6AB5C 21B86300 D9BC99F5 6E9DD1B7 F1DC4774

F.8.3 示例 2：域F_p，224 位素数p，SHA-224

F.8.3.1 参数

域为F_p，其中p的十六进制表示如下：

p = D7C134AA 26436686 2A183025 75D1D787 B09F0757 97DA89F5 7EC8C0FF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = 68A5E62C A9CE6C1C 299803A6 C1530B51 4E182AD8 B0042A59 CAD29F43

b = 2580F63C CFE44138 870713B1 A92369E3 3E2135D2 66DBB372 386C400B

G = (G_X, G_Y)

G_X = D9029AD2 C7E5CF43 40823B2A 87DC68C9 E4CE3174 C1E6EFDE E12C07D

G_Y = 58AA56F7 72C0726F 24C6B89E 4ECDAC24 354B9E99 CAA3F6D3 761402CD

q = D7C134AA 26436686 2A183025 75D0FB98 D116BC4B 6DDEBCA3 A5A7939F

F.8.3.2 签名密钥与验证密钥

X = 7E75BC2C D573B38A ED0977AD 611763DD 57FB29B2 20883344 B81DF037

Y = (Y_X, Y_Y)

Y_X = 8B29B268 866CEDCD A528F443 CAE7B07B F82BDC59 1A4BA29A C5E7BA4E

Y_Y = CD7746C1 4DEEA220 EA1BF164 C203C46E 60AF6699 CE6E1448 076B5807

F.8.3.3 逐消息数据

M = "brainpoolP224r1"的 ASCII 形式 =

62 72 61 69 6E 70 6F 6F 6C 50 32 32 34 72 31

K = 5B604F2C 35ED0401 FCA31E88 0CB55C2A 7456E71A 5CBAA8DF 2FC03CA9

Π = (Π_X, Π_Y)

Π_X = 60FBB2B1 5F055CD1 D482ED6D C5069C8F 624A3405 B67D11B3 B65E0234

Π_Y = 2C4359F0 A5A69F5A 29D1C1F3 86C1DCA6 5A47D160 DA1FBFB2 BA5A2FDB

$h(M)$ = AC2AC36A D5DAF131 951BA30B 330722C7 4BCFFF79 0617D1F0 908E06AF

F.8.3.4 签名

R = 60FBB2B1 5F055CD1 D482ED6D C5069C8F 624A3405 B67D11B3
B65E0234

S = 5A050F05 AF0B106B A3F14696 E6162CA4 6FBABD2C 144419DB
B5BFBDC0

F.8.3.5 验证

R' = 60FBB2B1 5F055CD1 D482ED6D C5069C8F 624A3405 B67D11B3
B65E0234

F.8.4 示例 3：域F_p，256 位素数p，SHA-256

F.8.4.1 参数

域为F_p，其中p的十六进制表示如下：

p = A9FB57DB A1EEA9BC 3E660A90 9D838D72 6E3BF623 D5262028
2013481D 1F6E5377

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = 7D5A0975 FC2C3057 EEF67530 417AFFE7 FB8055C1 26DC5C6C
E94A4B44 F330B5D9

b = 26DC5C6C E94A4B44 F330B5D9 BBD77CBF 95841629 5CF7E1CE
6BCCDC18 FF8C07B6

G = (G_X, G_Y)

G_X = 8BD2AEB9 CB7E57CB 2C4B482F FC81B7AF B9DE27E1 E3BD23C2
3A4453BD 9ACE3262

G_Y = 547EF835 C3DAC4FD 97F8461A 14611DC9 C2774513 2DED8E54
5C1D54C7 2F046997

q = A9FB57DB A1EEA9BC 3E660A90 9D838D71 8C397AA3 B561A6F7
901E0E82 974856A7

F.8.4.2 签名密钥与验证密钥

X = 52B929B4 0297437B 98973A2C 437E8F03 A231EB61 E0CD38FD
AD802F00 D55A13A3

Y = (Y_X, Y_Y)

Y_X = 90A53E95 D88397AC 76C7C128 297134D9 4BB52866 AD6474C1
3690A5E1 6848AC0D

Y_Y = 0C31F00F 0B3EAC60 A92A19AD 96E9BA31 3A43E100 D4D68FFF
2B8F1D8C D6790714

F.8.4.3 逐消息数据

M = "brainpoolP256r1"的 ASCII 形式 =

62 72 61 69 6E 70 6F 6F 6C 50 32 35 36 72 31

K = E6421272 DDAB9C20 7B119BDD 10C03861 005752EE ABB3AC97
513041AD E6286D9

Π = (Π_X, Π_Y)

Π_X = 829349E3 B6E1F3E5 15EB9581 BE0F958D CCAAA6B6 8D83BA77
01DD7A08 67E44EA7

Π_Y = 0E927978 F600F907 68B68C9E 572D33EC 2F8D8FC9 D577D743
8FDEE63D 27CE9763

$h(M)$ = DB7A981C 4E37DDE0 AEA27A34 E3179BD6 DF307204 75A7993A
FA93DF1D A7EC9910

F.8.4.4 签名

R = 829349E3 B6E1F3E5 15EB9581 BE0F958D CCAAA6B6 8D83BA77
01DD7A08 67E44EA7

S = 3DC2F103 296A793E 50DC2266 657470A4 0D2C9EA1 CA797DEA
610042B7 730BBDCE

F.8.4.5 验证

R' = 829349E3 B6E1F3E5 15EB9581 BE0F958D CCAAA6B6 8D83BA77
01DD7A08 67E44EA7

F.9 EC-RDSA 机制

F.9.1 示例 1：域F_p，256 位素数p，SHA-256

F.9.1.1 概述

在以下示例中，SHA-256 被专门用作哈希函数，哈希码即为 SHA-256 的输出值，并按照附录B转换为相应的数据项。

从安全角度来看，应避免使用密码学上弱的曲线（例如，应确保特定曲线不易受到椭圆曲线离散对数问题特殊实例攻击的影响）。

F.9.1.2 参数

域为F_p，其中p的十六进制表示如下：

p = 80000000 00000000 00000000 00000000 00000000 00000000
00000000 00000431

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = 7

b = 5FBFF498 AA938CE7 39B8E022 FBAFEF40 563F6E6A 3472FC2A
514C0CE9 DAE23B7E

G = (G_X, G_Y)

G_X = 2

G_Y = 08E2A8A0 E65147D4 BD631603 0E16D19C 85C97F0A 9CA26712
2B96ABBC EA7E8FC8

q = 80000000 00000000 00000000 00000001 50FE8A18 92976154
C59CFC19 3ACCF5B3

F.9.1.3 签名密钥与验证密钥

X = 7A929ADE 789BB9BE 10ED359D D39A72C1 1B60961F 49397EEE
1D19CE98 91EC3B28

Y = (Y_X, Y_Y)

Y_X = 7F2B49E2 70DB6D90 D8595BEC 458B50C5 8585BA1D 4E9B788F
6689DBD8 E56FD80B

Y_Y = 26F1B489 D6701DD1 85C8413A 977B3CBB AF64D1C5 93D26627
DFFB101A 87FF77DA

F.9.1.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

$h(M)$ = BA7816BF 8F01CFEA 414140DE 5DAE2223 B00361A3 96177A9C
B410FF61 F20015AD

K = 77105C9B 20BCD312 2823C8CF 6FCC7B95 6DE33814 E95B7FE6
4FED9245 94DCEAB3

Π = (Π_X, Π_Y)

Π_X = 41AA28D2 F1AB1482 80CD9ED5 6FEDA419 74053554 A42767B8
3AD043FD 39DC0493

Π_Y = 489C375A 9941A304 9E33B343 61DD2041 72AD98C3 E5916DE2
7695D22A 61FAE46E

F.9.1.5 签名

R = 41AA28D2 F1AB1482 80CD9ED5 6FEDA419 74053554 A42767B8
3AD043FD 39DC0493

S = 0A7BA472 2DA5693F 229D175F AB6AFB85 7EC2273B 9F88DA58
92CED311 7FCF1E36

F.9.1.6 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 41AA28D2 F1AB1482 80CD9ED5 6FEDA419 74053554 A42767B8
3AD043FD 39DC0493

Π'_Y = 489C375A 9941A304 9E33B343 61DD2041 72AD98C3 E5916DE2
7695D22A 61FAE46E

R' = 41AA28D2 F1AB1482 80CD9ED5 6FEDA419 74053554 A42767B8
3AD043FD 39DC0493

F.9.2 示例 2：域F_p，512 位素数p，SHA-512

F.9.2.1 概述

在以下示例中，SHA-512 被专门用作哈希函数，哈希码即为 SHA-512 的输出值，并按照附录B转换为相应的数据项。

从安全角度来看，应避免使用密码学上弱的曲线（例如，应确保特定曲线不易受到椭圆曲线离散对数问题特殊实例攻击的影响）。

F.9.2.2 参数

域为F_p，其中p的十六进·制表示如下：

p = 4531ACD1 FE0023C7 550D267B 6B2FEE80 922B14B2 FFB90F04
D4EB7C09 B5D2D15D F1D85274 1AF4704A 0458047E 80E4546D
35B8336F AC224DD8 1664BBF5 28BE6373

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = 7

b = 1CFF0806 A31116DA 29D8CFA5 4E57EB74 8BC5F377 E49400FD
D788B649 ECA1AC43 61834013 B2AD7322 480A89CA 58E0CF74
BC9E540C 2ADD6897 FAD0A308 4F302ADC

G = (G_X, G_Y)

G_X = 24D19CC6 4572EE30 F396BF6E BBFD7A6C 5213B3B3 D7057CC8
25F91093 A68CD762 FD606112 62CD838D C6B60AA7 EEE804E2
8BC84997 7FAC33B4 B530F1B1 20248A9A

G_Y = 2BB312A4 3BD2CE6E 0D020613 C857ACDD CFBF061E 91E5F2C3
F32447C2 59F39B2C 83AB156D 77F1496B F7EB3351 E1EE4E43
DC1A18B9 1B24640B 6DBB92CB 1ADD371E

q = 4531ACD1 FE0023C7 550D267B 6B2FEE80 922B14B2 FFB90F04
D4EB7C09 B5D2D15D A82F2D7E CB1DBAC7 19905C5E ECC423F1
D86E25ED BE23C595 D644AAF1 87E6E6DF

F.9.2.3 签名密钥与验证密钥

X = 0BA6048A ADAE241B A40936D4 7756D7C9 3091A0E8 51466970
0EE7508E 508B1020 72E8123B 2200A056 3322DAD2 827E2714
A2636B7B FD18AADF C6296782 1FA18DD4

Y = (Y_X, Y_Y)

Y_X = 115DC5BC 96760C7B 48598D8A B9E740D4 C4A85A65 BE33C181
5B5C320C 854621DD 5A515856 D13314AF 69BC5B92 4C8B4DDF
F75C4541 5C1D9DD9 DD33612C D530EFE1

Y_Y = 37C7C90C D40B0F56 21DC3AC1 B751CFA0 E2634FA0 503B3D52
639F5D7F B72AFD61 EA199441 D943FFE7 F0C70A27 59A3CDB8
4C114E1F 9339FDF2 7F35ECA9 3677BEEC

F.9.2.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

$h(M)$ = DDAF35A1 93617ABA CC417349 AE204131 12E6FA4E 89A97EA2
0A9EEEE6 4B55D39A 2192992A 274FC1A8 36BA3C23 A3FEEBBD
454D4423 643CE80E 2A9AC94F A54CA49F

K = 3B109D0F 05D95496 1A085730 483EEC3A 8A544589 0E76066E
2EE0410C 33C1EE1E 8D86B971 6CB12FD8 F91843C2 C36C82A4
E29FFF5E BCEF22CD E4062389 76F28E85

Π = (Π_X, Π_Y)

Π_X = 13C56557 E300898B F6C91A08 AF0CAF80 1046A2DC 58CF7E84
A15DA3B6 89C0EB29 73F5BE70 27DBDD77 BCE5D337 6AD5793C
21315785 AA6D2536 A20C9158 14F2ADDC

Π_Y = 308AB8AE 5DF67642 7A94C9FC 014CC352 267F3DC4 48003E1C
491768DE 7F660A00 CF4EB1B7 BC67C0CF E7D20256 B84F690C
BB5751A6 A6328140 1287C279 BB96F231

F.9.2.5 签名

R = 13C56557 E300898B F6C91A08 AF0CAF80 1046A2DC 58CF7E84
A15DA3B6 89C0EB29 73F5BE70 27DBDD77 BCE5D337 6AD5793C
21315785 AA6D2536 A20C9158 14F2ADDC

S = 32C0B15B E367583B B3FAEFF1 49AF87D1 18BF18E1 3487E0C6
AB7580B8 62EC104A 41EC9A5F B17B0E0E DBCFFD92 0D6F627E
704A82CC 534127F6 44FDC958 984DDCA0

F.9.2.6 验证

$\Pi' = (\Pi'_X, \Pi'_Y)$

Π'_X = 13C56557 E300898B F6C91A08 AF0CAF80 1046A2DC 58CF7E84
A15DA3B6 89C0EB29 73F5BE70 27DBDD77 BCE5D337 6AD5793C
21315785 AA6D2536 A20C9158 14F2ADDC

Π'_Y = 308AB8AE 5DF67642 7A94C9FC 014CC352 267F3DC4 48003E1C
491768DE 7F660A00 CF4EB1B7 BC67C0CF E7D20256 B84F690C
BB5751A6 A6328140 1287C279 BB96F231

R' = 13C56557 E300898B F6C91A08 AF0CAF80 1046A2DC 58CF7E84
A15DA3B6 89C0EB29 73F5BE70 27DBDD77 BCE5D337 6AD5793C
21315785 AA6D2536 A20C9158 14F2ADDC

F.10 EC-SDSA 机制

F.10.1 示例 1：域F_p，256 位素数p，SHA-256

F.10.1.1 概述

本示例使用 NIST 曲线 P256 作为椭圆曲线。SHA-256 被专门用作哈希函数，哈希码即为 SHA-256 的输出值，并按照附录B转换为相应的数据项。

从安全角度来看，应避免使用密码学上弱的曲线（例如，应确保特定曲线不易受到椭圆曲线离散对数问题特殊实例攻击的影响）。

F.10.1.2 参数

域为F_p，其中p的十六进制表示如下：

p = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF
FFFFFFFF FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF
FFFFFFFF FFFFFFFC

b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6
3BCE3C3E 27D2604B

$G = (G_X, G_Y)$

G_X = 6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0
F4A13945 D898C296

G_Y = 4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE
CBB64068 37BF51F5

q = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84

F3B9CAC2 FC632551

F.10.1.3 签名密钥与验证密钥

X = 5202A3D8 ACAF6909 D12C9A77 4CD886F9 FBA61137 FFD3E8E7 6AED363F B47AC492

$Y = (Y_X, Y_Y)$

Y_X = 09B58B88 323C52D1 080AA525 C89E8E12 C6F40FCB 014640FA 88081ED9 E9352DE7

Y_Y = 5CCBBD18 95385162 38B0B0B2 8ACB5F0B 5E27217C 3A987242 1219DE0A EEBF1080

F.10.1.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = DE7E0E5E 663F2418 3414B7C7 2F24546B 81E9E5F4 10BEBF26 F3CA5FA8 2F5192C8

$\Pi = (\Pi_X, \Pi_Y)$

Π_X = 847CE3CD 474FEC19 722AA9BA 81AFBF34 7EE2D70E D067413F 1F716783 27A758CA

Π_Y = DBFAD4AF 8C1D93AB 9C16467E 96BD11B5 33643AA6 63498D8F 95919C6C A1AD91FC

$\text{FE2BS}(r, \Pi_X)\|\text{FE2BS}(r, \Pi_Y)$ = 847CE3CD 474FEC19 722AA9BA 81AFBF34 7EE2D70E D067413F 1F716783 27A758CA DBFAD4AF 8C1D93AB 9C16467E 96BD11B5 33643AA6 63498D8F 95919C6C A1AD91FC

$R = h(\text{FE2BS}(r, \Pi_X)\|\text{FE2BS}(r, \Pi_Y)\|M)$ = 5A79A0AA 9B241E38 1A594B22 0554D096 A5F09FA6 28AD9A33 C3CE4393 ADE1DEF7

对于 EC-SDSA 的优化变体，

$R = h(\text{FE2BS}(r, \Pi_X)\|M)$ = D7FB8135 D8EA45E8 FB3C9059 F146E263 0EF4BD51 C4006A92 EDB4C8B0 849963FB

F.10.1.5 签名

R = 5A79A0AA 9B241E38 1A594B22 0554D096 A5F09FA6 28AD9A33 C3CE4393 ADE1DEF7

S = 5C0EB78B 67A513C3 E53B2619 F96855E2 91D5141C 7CD0915E 1D04B347 457C9601

对于 EC-SDSA 的优化变体，

R = D7FB8135 D8EA45E8 FB3C9059 F146E263 0EF4BD51 C4006A92 EDB4C8B0 849963FB

S = B46D1525 379E02E2 32D97928 265B7254 EA2ED978 13454388 C1A08F62 DCCD70B3

F.10.1.6 验证

$\Pi' = (\Pi'_X, \Pi'_Y)$

Π'_X = 847CE3CD 474FEC19 722AA9BA 81AFBF34 7EE2D70E D067413F
1F716783 27A758CA

Π'_Y = DBFAD4AF 8C1D93AB 9C16467E 96BD11B5 33643AA6 63498D8F
95919C6C A1AD91FC

$\text{FE2BS}(r,\Pi'_X)\|\text{FE2BS}(r,\Pi'_Y)$ = 847CE3CD 474FEC19 722AA9BA 81AFBF34
7EE2D70E D067413F 1F716783 27A758CA DBFAD4AF 8C1D93AB
9C16467E 96BD11B5 33643AA6 63498D8F 95919C6C A1AD91FC

$R' = h(\text{FE2BS}(r,\Pi'_X)\|\text{FE2BS}(r,\Pi'_Y)\|M)$ = 5A79A0AA 9B241E38 1A594B22
0554D096 A5F09FA6 28AD9A33 C3CE4393 ADE1DEF7

对于 EC-SDSA 的优化变体，

$R' = h(\text{FE2BS}(r,\Pi'_X)\|M)$ = D7FB8135 D8EA45E8 FB3C9059 F146E263
0EF4BD51 C4006A92 EDB4C8B0 849963FB

F.10.2 示例 2：域F_p，384 位素数p，SHA-384

F.10.2.1 概述

本示例使用 NIST 曲线 P384 作为椭圆曲线。SHA-384 被专门用作哈希函数，哈希码即为 SHA-384 的输出值，并按照附录B转换为相应的数据项。

从安全角度来看，应避免使用密码学上弱的曲线（例如，应确保特定曲线不易受到椭圆曲线离散对数问题特殊实例攻击的影响）。

F.10.2.2 参数

域为F_p，其中p的十六进制表示如下：

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFC

b = B3312FA7 E23EE7E4 988E056B E3F82D19 181D9C6E FE814112
0314088F 5013875A C656398D 8A2ED19D 2A85C8ED D3EC2AEF

G = (G_X, G_Y)

G_X = AA87CA22 BE8B0537 8EB1C71E F320AD74 6E1D3B62 8BA79B98
59F741E0 82542A38 5502F25D BF55296C 3A545E38 72760AB7

G_Y = 3617DE4A 96262C6F 5D9E98BF 9292DC29 F8F41DBD 289A147C
E9DA3113 B5F0B8C0 0A60B1CE 1D7E819D 7A431D7C 90EA0E5F

q = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
C7634D81 F4372DDF 581A0DB2 48B0A77A ECEC196A CCC52973

F.10.2.3 签名密钥与验证密钥

X = 7E4914FE 4B617089 F9FE80AD 913D5530 EEC4F30B CA1AD66E
7B5AEACF 29D2F567 D9A8F4A4 552A1A1A F3E0B6D0 A49DFCC8

Y = (Y_X, Y_Y)

Y_X = 16C000E5 92FADA35 2CB15605 DA2AF639 67B23F9E 9D36F324
0374D357 BB3B7D04 CB76382C 707B18FE 104B632E E1860C36

Y_Y = 0CBB8CF8 65CFB286 B0009342 CA2799EF 2BE3D040 806BF237
88DEE3F8 2E0C2006 7DCD474F 202B4C34 2C4F36BF 6B84E197

F.10.2.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = 8A29E772 357BBA6F 5C9EA765 D5082B9B C7A74C33 E9D94D49
FB2C9D3B 523A8216 9682ECF1 6F1D0626 9042F3AF 044B4DE8

Π = (Π_X, Π_Y)

Π_X = 896BF1AC 124A3DBC FF82705E D7699A45 0C827103 47DEE91C
7D061D4B C512ADE1 2D2660B9 BA0F6615 C9324E0A 5E3C1AE7

Π_Y = B0B11AEF 22E5348D B2E3E106 99215B6F 4F87D11B EC2CBDDB
F971A123 3BAD21A2 18DFE905 C0010D88 E7D64625 2633620E

$\text{FE2BS}(r, \Pi_X)\|\text{FE2BS}(r, \Pi_Y)$ = 896BF1AC 124A3DBC FF82705E D7699A45
0C827103 47DEE91C 7D061D4B C512ADE1 2D2660B9 BA0F6615
C9324E0A 5E3C1AE7 B0B11AEF 22E5348D B2E3E106 99215B6F
4F87D11B EC2CBDDB F971A123 3BAD21A2 18DFE905 C0010D88
E7D64625 2633620E

$R = h(\text{FE2BS}(r, \Pi_X)\|\text{FE2BS}(r, \Pi_Y)\|M)$ = F907553B B5C7DE02 9A2A5670
78DFF9B8 03EC6496 0D75BA73 A85590AC C0AC4479 AC52E51D
5691FCB0 69DC5CD2 4E0BCEC7

对于 EC-SDSA 的优化变体，

$R = h(\text{FE2BS}(r, \Pi_X)\|M)$ = 27D2F5B9 62A3ACF6 390A4718 EA540DA7
9612A60E AA15BEBB 00B9E166 5783F7C7 91CCAC42 2CEE815A
9C5DA367 8AC8D1F0

F.10.2.5 签名

R = F907553B B5C7DE02 9A2A5670 78DFF9B8 03EC6496 0D75BA73
A85590AC C0AC4479 AC52E51D 5691FCB0 69DC5CD2 4E0BCEC7

S = 0B9D66D5 DE70FAA8 B35634A3 7B33C2C4 60B8DC0B D4C8745B
B84DC15C A8570B07 9258F977 DA8B4061 F3DA6EBD 7C429A89

对于 EC-SDSA 的优化变体，

R = 27D2F5B9 62A3ACF6 390A4718 EA540DA7 9612A60E AA15BEBB
00B9E166 5783F7C7 91CCAC42 2CEE815A 9C5DA367 8AC8D1F0

S = 22CC89CE B9E6BE84 15CC14B3 99BC66E6 F3A21E5B A38E09A6
DE8DE670 A145C0E4 74D5CC88 BE8878F0 123CC662 25A1BA12

F.10.2.6 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 896BF1AC 124A3DBC FF82705E D7699A45 0C827103 47DEE91C
7D061D4B C512ADE1 2D2660B9 BA0F6615 C9324E0A 5E3C1AE7

Π'_Y = B0B11AEF 22E5348D B2E3E106 99215B6F 4F87D11B EC2CBDDB
F971A123 3BAD21A2 18DFE905 C0010D88 E7D64625 2633620E

$\text{FE2BS}(r, \Pi'_X)\|\text{FE2BS}(r, \Pi'_Y)$ = 896BF1AC 124A3DBC FF82705E D7699A45

0C827103 47DEE91C 7D061D4B C512ADE1 2D2660B9 BA0F6615
C9324E0A 5E3C1AE7 B0B11AEF 22E5348D B2E3E106 99215B6F
4F87D11B EC2CBDDB F971A123 3BAD21A2 18DFE905 C0010D88
E7D64625 2633620E

$R' = h(\text{FE2BS}(r, \Pi'_X) \| \text{FE2BS}(r, \Pi'_Y) \| M)$ = F907553B B5C7DE02 9A2A5670
78DFF9B8 03EC6496 0D75BA73 A85590AC C0AC4479 AC52E51D
5691FCB0 69DC5CD2 4E0BCEC7

对于 EC-SDSA 的优化变体，

$R' = h(\text{FE2BS}(r, \Pi'_X) \| M)$ = 27D2F5B9 62A3ACF6 390A4718 EA540DA7
9612A60E AA15BEBB 00B9E166 5783F7C7 91CCAC42 2CEE815A
9C5DA367 8AC8D1F0

F.11 EC-FSDSA 机制

F.11.1 示例 1：域F_p，256 位素数p，SHA-256

F.11.1.1 概述

本示例使用 NIST 曲线 P256 作为椭圆曲线。SHA-256 被专门用作哈希函数，哈希码即为 SHA-256 的输出值，并按照附录B转换为相应的数据项。

从安全角度来看，应避免使用密码学上弱的曲线（例如，应确保特定曲线不易受到椭圆曲线离散对数问题特殊实例攻击的影响）。

F.11.1.2 参数

域为F_p，其中p的十六进制表示如下：

p = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF
FFFFFFFF FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF
FFFFFFFF FFFFFFFC

b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6
3BCE3C3E 27D2604B

G = (G_X, G_Y)

G_X = 6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0
F4A13945 D898C296

G_Y = 4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE
CBB64068 37BF51F5

q = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84
F3B9CAC2 FC632551

F.11.1.3 签名密钥与验证密钥

X = ACCA7F0D D3AC535F 489B340F 6BD7F503 61B0EE09 5AE6A228
9A6AB329 238123E5

Y = (Y_X, Y_Y)

Y_X = B54E30D3 72FFB75C 0A5E495C 59A036BE ABB54134 00F0ADF3

C2B5B160 BA959578

Y_Y = 58F8505E 1673AA64 51F84C37 BF338519 108AA89E 33AEA991
5168D6F1 E3B67E13

F.11.1.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = 894DEAB4 4D88450F E8DAC663 F0E58650 31E875BA 224C0601
3C53D0E3 0109C207

Π = (Π_X, Π_Y)

Π_X = AF312FBD 7792125C 5CDFBA69 E6D36990 0ACE9A70 BA653FFF
BD9140E0 0079FAE8

Π_Y = B7CEC570 16A0B97A A069D54E 0DA95E45 FB50B677 1FB69F53
FEF00FC8 B00E1FEC

R = AF312FBD 7792125C 5CDFBA69 E6D36990 0ACE9A70 BA653FFF
BD9140E0 0079FAE8
B7CEC570 16A0B97A A069D54E 0DA95E45 FB50B677 1FB69F53
FEF00FC8 B00E1FEC

$h(R\|M)$ = 44E762CA 7B26973F CDFA1301 0658E68E 88EA9D06
FB76782D AFC0EB78 1F9213C2

F.11.1.5 签名

R = AF312FBD 7792125C 5CDFBA69 E6D36990 0ACE9A70 BA653FFF
BD9140E0 0079FAE8 B7CEC570 16A0B97A A069D54E 0DA95E45
FB50B677 1FB69F53 FEF00FC8 B00E1FEC

S = 25847040 2304BC2D B44F3B2A 20C08FF2 A64F566B AA2EB7BF
37E1619B 6AE09844

F.11.1.6 验证

Π' = (Π'_X, Π'_Y)

Π'_X = AF312FBD 7792125C 5CDFBA69 E6D36990 0ACE9A70 BA653FFF
BD9140E0 0079FAE8

Π'_Y = B7CEC570 16A0B97A A069D54E 0DA95E45 FB50B677 1FB69F53
FEF00FC8 B00E1FEC

R' = AF312FBD 7792125C 5CDFBA69 E6D36990 0ACE9A70 BA653FFF
BD9140E0 0079FAE8
B7CEC570 16A0B97A A069D54E 0DA95E45 FB50B677 1FB69F53
FEF00FC8 B00E1FEC

F.11.2 示例 2：域F_p，384 位素数p，SHA-384

F.11.2.1 概述

本示例使用 NIST 曲线 P384 作为椭圆曲线。SHA-384 被专门用作哈希函数，哈希码即为 SHA-384 的输出值，并按照附录B转换为相应的数据项。

从安全角度来看，应避免使用密码学上弱的曲线（例如，应确保特定曲线不易受到椭圆曲线离散对数问题特殊实例攻击的影响）。

F.11.2.2 参数

域为F_p，其中p的十六进制表示如下：

p = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFF

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

a = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFC

b = B3312FA7 E23EE7E4 988E056B E3F82D19 181D9C6E FE814112
0314088F 5013875A C656398D 8A2ED19D 2A85C8ED D3EC2AEF

G = (G_X, G_Y)

G_X = AA87CA22 BE8B0537 8EB1C71E F320AD74 6E1D3B62 8BA79B98
59F741E0 82542A38 5502F25D BF55296C 3A545E38 72760AB7

G_Y = 3617DE4A 96262C6F 5D9E98BF 9292DC29 F8F41DBD 289A147C
E9DA3113 B5F0B8C0 0A60B1CE 1D7E819D 7A431D7C 90EA0E5F

q = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
C7634D81 F4372DDF 581A0DB2 48B0A77A ECEC196A CCC52973

F.11.2.3 签名密钥与验证密钥

X = 95A4D257 A7298C66 10A37558 785036DE 6F9FB997 735C076B
8C8A18B2 AAAC3142 507A2560 3D7C95F9 E5F0307E C5A56D7E

Y = (Y_X, Y_Y)

Y_X = 75E4FB35 F5FBBE88 5AC414BA 51C2F9F7 57D071F8 54A839A5
84031158 4A0559A7 2CDC563E 29D752AB 4D511882 7489F434

Y_Y = 2A693A17 2EC9F108 BE348528 8695EF4E 6B20971F 69859E9D
13044881 FCD76224 C6369A09 B96DF415 4592ADB3 083A2005

F.11.2.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = 51C5B8B2 E59CF78F 54E77CDB 0B2E2669 B66B3284 8B7B5378
01483DE2 39422745 6F4930C8 53FBFF6C 58FA6E1C C5D97466

Π = (Π_X, Π_Y)

Π_X = 29B785AB 7FAC1F80 F64CE2F2 D88ABA8F E6103B25 65FEB5AE
82FB6CF5 8F8CE1E7 3F7A8D68 3BEABD24 B7EA78F6 013C9F70

Π_Y = DE94FAD1 7281F4A6 0AA25423 E9D87122 96540219 5239B839
FCD44CDF 545BF74E 1300C9F5 7FFC830D EFD97B66 D57E9D07

R = 29B785AB 7FAC1F80 F64CE2F2 D88ABA8F E6103B25 65FEB5AE
82FB6CF5 8F8CE1E7 3F7A8D68 3BEABD24 B7EA78F6 013C9F70
DE94FAD1 7281F4A6 0AA25423 E9D87122 96540219 5239B839
FCD44CDF 545BF74E 1300C9F5 7FFC830D EFD97B66 D57E9D07

$h(R\|M)$ = 970BC355 820ADDDD 8F39CBD9 E386BE70 143B965F B4E720AF
1E31658F 87410729 758DEEDD B706221C 332308A6 1D9D0F40

F.11.2.5 签名

R = 29B785AB 7FAC1F80 F64CE2F2 D88ABA8F E6103B25 65FEB5AE
82FB6CF5 8F8CE1E7 3F7A8D68 3BEABD24 B7EA78F6 013C9F70
DE94FAD1 7281F4A6 0AA25423 E9D87122 96540219 5239B839
FCD44CDF 545BF74E 1300C9F5 7FFC830D EFD97B66 D57E9D07

S = D1858062 C5504E21 78523926 423FDD83 99A8BA2B 85BF4585
3F8E04BF 20441516 E71A78B0 9C7A7EE6 20B7F537 E6C1DEEE

F.11.2.6 验证

$\Pi' = (\Pi'_X, \Pi'_Y)$

Π'_X = 29B785AB 7FAC1F80 F64CE2F2 D88ABA8F E6103B25 65FEB5AE
82FB6CF5 8F8CE1E7 3F7A8D68 3BEABD24 B7EA78F6 013C9F70

Π'_Y = DE94FAD1 7281F4A6 0AA25423 E9D87122 96540219 5239B839
FCD44CDF 545BF74E 1300C9F5 7FFC830D EFD97B66 D57E9D07

R' = 29B785AB 7FAC1F80 F64CE2F2 D88ABA8F E6103B25 65FEB5AE
82FB6CF5 8F8CE1E7 3F7A8D68 3BEABD24 B7EA78F6 013C9F70
DE94FAD1 7281F4A6 0AA25423 E9D87122 96540219 5239B839
FCD44CDF 545BF74E 1300C9F5 7FFC830D EFD97B66 D57E9D07

F.12 IBS-1 机制

F.12.1 示例 1：域F_p，512 位素数p，SHA-1

F.12.1.1 概述

在本示例中，F_p^2中的元素表示为$a + \sigma b$，其中a和b为F_p中的元素，σ为F_p^2中满足$\sigma^2 + 1 \equiv 0 \bmod p$的元素。将一个挠群中的点转换为另一个挠群中的元素（以实现配对的非退化性）的畸变映射定义如下：$\phi P: (P_X, P_Y) \rightarrow (\beta P_X, P_Y)$，其中$\beta \neq 1$为$F_p^2$中满足$\beta^3 - 1 \equiv 0 \bmod p$的元素。$<P, Q>$实现为两个输入点$P$和$\phi(Q)$上的约化 Tate 配对。

在本示例中，H_2实现为 SHA-1。H_1对给定消息M的实现为：$H_1(M) = \text{I2P}(\text{BS2I}(512, \text{MGF1}(M)))$，其中 MGF1(M)为
SHA-1(M||0)||SHA-1(M||1)||SHA-1(M||2)||SHA-1(M||3)的最左侧 512 位。此处，常量"0"、"1"、"2"和"3"应作为 32 位整数处理以匹配数据。

F.12.1.2 参数

域为F_p，其中p的十六进制表示如下：

p = B35FA5FD E47FA1AB BB1E57E9 3BA1FF96 38B89B99 5C49BE81 A38E3194
A0983816 4EE51FB9 1D285832 F9A05D63 9C8D9680 10C93A35 27E561F2
FD6A45CC 70ABA1FB

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + 1$。

q = 80000000 000FFFFF FFFFFFFF FFFFFFFF FFFFFFFF

$\beta = (\beta_a, \beta_b)$

β_a = 59AFD2FE F23FD0D5 DD8F2BF4 9DD0FFCB 1C5C4DCC AE24DF40 D1C718CA
504C1C0B 27728FDC 8E942C19 7CD02EB1 CE46CB40 08649D1A 93F2B0F9
7EB522E6 3855D0FD

β_b = 20C3F863 7D0F88AE 295531C6 82537070 5EFA0F99 39D1CDE4 F58A25F4

3722E229 5410CA05 B2360EA8 625C8203 626C5254 3E63404A B83448B0
3FDA177A 7A2DDB21

$P = (P_X, P_Y)$

P_X = 7EE6F118 9329ADB4 1E8CD405 2295F1A7 6096631D F065DD38 85FFF26B
8ED52022 7BFC3F3D 07FE9CFE 093424BA 9DBD4C0D 73FFF367 3CE2C922
F0B73C50 2992093B

P_Y = 67C855C9 E6B617BE B24B792A E9C3E21E 95F37006 25B91058 3BC0B293
C36A762C FEBA4266 038989CC 59797235 C6116D99 8A97B805 FF82C664
53720E5C 1DE95E3E

U = 50BD36F5 967F41F4 93B1F667 4AD462F6 2B7D69D8

$V = [U]P = (V_X, V_Y)$

V_X = 04863127 7E2F50F6 CCCD235D EC856E8C FB22F17A 232D6229 7B7ABA44
77639ADE F2CC4863 5C19ED50 9E7F3F2C C5EF1F5B 5225E45D 30C269A3
97978783 D7E0DD86

V_Y = 75651DE4 8D8AEC70 10B4084A 05EE2EFB 0D72CF41 4F5B6113 539198F5
E800394F DC2317AA 3B6DEE3E E800394F 470020FB E6EE9811 CDDC5991
4BD0CB38 2E203F01 0816E8B7

$\phi(V) = (\beta V_X, V_Y)$

$(\beta V_X)_a$ = B11C8D6A 2567F930 54B7C63A 455F484F BB2722DC 4AB30D6C E5D0D472
64E66AA6 D57EFB87 6F1B618A AA60BDCD 399606D2 67B64806 8F842D21
319E820A 84BB3338

$(\beta V_X)_b$ = AA701CC7 9EF755C5 5E33269C 4F54BFB2 819E0F5A 6F6C8084 557428CA
71F0A225 393CB501 66D2456C A6F1D22A C983D4FF CBF03583 813D2893
7087739B 7E279447

V_Y = 75651DE4 8D8AEC70 10B4084A 05EE2EFB 0D72CF41 4F5B6113 539198F5
DC2317AA 3B6DEE3E E800394F 470020FB E6EE9811 CDDC5991 4BD0CB38
2E203F01 0816E8B7

F.12.1.3 签名密钥与验证

ID = “alice@network.com”

$MGF1(ID)$ = 85499895 C3B4F412 56E84BDC E3D76405 33612345 D6BF0725 B7D0391
D53A680B 2DD96646 A32B244E DA614B5D 8E989340 473B7885 F243CE0
0FBD2661 43B85636

$\mathrm{I2P}(\mathrm{BS2I}(512, \mathrm{MGF1}(ID))) = (Y_X, Y_Y)$

Y_X = 8BB6EFE1 AF00D035 D0B286F1 94BC3128 A639949A B9D06405 95F837A
E131C3C6 B37F2905 4D58FB21 9FA190B7 A65BE61F 4A6945B5 4A1A36C
6166F047 2F63D303

Y_Y = 3D605915 84CF4C40 2E81145C 0047D9BE 2733A3C2 66F7A8F8 CA2F50C
635424C3 8AFF64FA F9B5CD04 4AD0236F BC7BEB72 0D1909DA CC3C3FE
3022CDB7 7946F37D

$X = [U]Y = (X_X, X_Y)$

X_X = B26A00EA 12877F1D 049DC4F2 4030CDB1 632AA9FA D48E209D CC1A116
A159CBDD 540F417D FC584FF9 4C959E64 3BFA2131 7EB54356 BB8650B
8803CB32 F607C87B

X_Y = 3527A3A6 E22290F9 0CC5ECBE F6CFE3DA F0F7EF16 327606C8 2456C9F
6C7A2818 84832B75 41D8EAAE DB1B0FDA 78335B84 C56FEF1F 88423C3
1393760B AE4520D1

F.12.1.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = 3846BAA7 C64DA228 E356EE28 5E2DE770 6D2C4F83

Π = (Π_a, Π_b)

Π_a = 0035A456 EB8285BA CC8912FD 84BA4A9A 68CEA16A D69570FC 5E10C90D
44D84269 2F3097A6 A715AE83 C48E0413 EFEACBA1 A6BAF06D 621B5835
E441C245 4C9601ED

Π_b = 8302C6D9 5413BAAE E5484202 8189CA1F BDB85BA2 D1544428 05158303
ECEE99B4 6A24CCD2 D8580A1F 00A9B2A2 568125C3 DE7925F6 0E1FC61D
F78F9F1D E7A6C76D

F.12.1.5 签名

R = $\mathrm{BS2I}(\gamma, H_2(M_1 \| \mathrm{FE2BS}(r, \Pi_a) \| \mathrm{FE2BS}(r, \Pi_b))) \bmod q$ =

01D7ED22 EC366550 D4246317 1662B5CA 1742C1E8

S = (S_X, S_Y)

S_X = 8E911C5B 986E0547 E5F31C57 1FDC384D F8968374 23B34BF6 EE8865C1
2BC1D56B 9E6D4B8A 15DAD877 685CB024 1FA07012 EC0E9183 B18955D5
570DAEBB BF0CD364

S_Y = 00CBFC08 18782909 2F6F2496 D1F2299E 5159B89F 6535E3E4 D708D77D
8FD7F262 E4E05331 81BD4933 A0CFA2FC 710ECC59 99078706 BA0B4F3A
468B61DA 3045A0E8

F.12.1.6 验证

R' = 01D7ED22 EC366550 D4246317 1662B5CA 1742C1E8

F.12.2 示例 2：域F_p，512 位素数p，SHA-1

F.12.2.1 概述

在本示例中，F_p^2中的元素表示为$a + \sigma b$，其中a和b为F_p中的元素，σ为F_p^2中满足$\sigma^2 + 1 \equiv 0 \bmod p$的元素。将一个挠群中的点转换为另一个挠群中的元素的畸变映射定义如下：$\phi P: (P_X, P_Y) \rightarrow (-P_X, \sigma P_Y)$。$<P, Q>$实现为两个输入点$P$和$\phi(Q)$上的约化 Tate 配对。

在本示例中，H_2实现为 SHA-1。H_1对给定消息M的实现为：$H_1(M) = \mathrm{I2P}(\mathrm{BS2I}(512, \mathrm{MGF1}(M)))$，其中 MGF1(M)为取 SHA-1(M||0)||SHA-1(M||1)||SHA-1(M||2)||SHA-1(M||3)的前 512 位并将首位置零的结果。

F.12.2.2 参数

域为F_p，其中p的十六进制表示如下：

p = 80000000 00000000 00000000 00000000 00020001 40000000 00000000
00000000 00000000 00010000 80000002 00000000 00000000 00000000
00000000 00080003

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + X$

$\#E$ = 80000000 00000000 00000000 00000000 00020001 40000000 00000000
00000000 00000000 00010000 80000002 00000000 00000000 00000000
00000000 00080004

q = 80000000 00000000 00000000 00000000 00020001

P = (P_X, P_Y)

P_X = 0DB4E0F7 22DD090D A2B6D8FE ADAF21D9 546AB265 1515AF9B A87108F3
4E1AE0E3 EB132C10 81452CC1 E52BB2A7 4287A0CB D8FF8DD9 3A225641
5321F0E4 C8892A50

P_Y = 762C096C 49F1AB04 7D7F37DE 537A4E7C 2991C400 22E0C9A9 B3F58B1B
9dF4F28A 4A4330E2 170E14D2 F55A0719 8B667D0B 01E5A482 3F07E921
8516481E 641970AC

U = 5C1A6406 3FCC6EA9 DAF85736 ECE7C438 688F6FA6

V = $[U]P = (V_X, V_Y)$

V_X = 6B8F666B CF6B4672 D4634753 1F734E71 41BCD5FD 125F3EF3 714EDC28
F6426900 75FFB5F7 9E745CC0 FB03F940 3BDCEFE8 ACBE6286 D5D9955C
2A0E5ED7 657748C6

V_Y = 69584E47 F3070FED 9800D6CD E0F314B4 03955126 1C5BFEF6 F3595F94
5958F7D9 34DCBD3D 63125410 CCD363F8 02DF1C7E 4A3D7AC7 24CF3865
0FB16EC1 7BB30A85

$\phi(V)$ = $(-V_X, V_Y)$

$-V_X$ = 14709994 3094B98D 2B9CB8AC E08CB18E BE452A04 2DA0C10C 8EB123D7
09BD96FF 8A004A08 618CA33F 84FC06C1 C4231017 53419D79 2A266AA3
D5F1A128 9A90B73D

F.12.2.3 签名密钥与验证密钥

ID = “alice@network.com”

$MGF1(ID)$ = 05499895 C3B4F412 56E84BDC E3D76405 33612345 D6BF0725 B7D0391
D53A680B 2DD96646 A32B244E DA614B5D 8E989340 473B7885 F243CE0
0FBD2661 43B85636

$\mathrm{I2P}(\mathrm{BS2I}(512, \mathrm{MGF1}(ID))) = (Y_X, Y_Y)$

Y_X = 28C1852E 0D0D6AF3 A55F7846 01D2B1B6 3CA8E9B1 597BB19A A7D2CCE
2EED68D9 356BEE5D 81B14B2D 095C6B6C 30461DCE A183BC61 A115FCE
D4DAC9AE 6EB92570

Y_Y = 1C417DF1 D1BB732F 6849AA30 F954042A 142901C2 6A0ED564 310D3A5
0A6EAC35 3855DDC0 485311E1 A12735E1 95D76E34 063FF0F5 DA251F3
E261AC66 D7BFA23B

X = $[U]Y = (X_X, X_Y)$

X_X = 494661E1 3FC73652 E6F9A500 CC9009AC D0B2085D 93709163 E186185
7E61F889 E9B5CAC9 87D23C51 95E3C3DE B4055DA6 BE911267 8CB481F
B115534A EBD6EF0B

X_Y = 06A03100 C34D09DD 9155DE07 61BED764 744ACA6C 6DC9169C EC9E245
2B058DFD 9D426B73 9E444976 7DB80539 C7B6C7A2 65B1B7EA 079A704
3949B35D 4AD88F24

F.12.2.4 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = 0EC982DA 1CAB8A86 692448F4 EB841BD6 CACB63E5

Π = (Π_a, Π_b)

Π_a = 647BA1C8 F922204C 5D0F4AD3 DD0E093B FF068454 D67B803C 82C0F082
876BC786 748E005C EBF2C2DA 67E2CA31 0CE29DED 0EFAF684 3C1660D0
549989A5 A2F46DF8

Π_b = 68DC5B24 5DB19947 E3ABA4C5 0AE4C90F 4C2B5AD5 F55DDFAB 61D46A5E
7BFA66B3 32431664 E1268632 159F22B6 5F745302 5FFD9E93 1314D23A
69D6E89D 759CF6BF

F.12.2.5 签名

R = $\mathrm{BS2I}(\gamma, H_2(M_1 \| \mathrm{FE2BS}(r, \Pi_a) \| \mathrm{FE2BS}(r, \Pi_b))) \bmod q$ =

06CD1062 D7B840D5 8B6973C5 7B075BF9 7877680F

S = (S_X, S_Y)

S_X = 6B59A8AB DB1752A5 46EC4F7E 8CB5FC6C 129BEEF3 726036CF 9CA55AE6
04A8EC16 05E987D9 C7234D7E 2D08A57A 42D79861 F350B298 51C3EF24
1331CEC7 B92FA579

S_Y = 4DD608CC 0A09C606 DFEFC33B 190F2CA5 E29E8811 49A54439 E437D8F0
91535531 30B03EE2 21322120 B6AE5964 9D0A11E7 492B0C32 A041029A
ED9E354E 03DBD353

F.12.2.6 验证

Π' = $(\Pi' a, \Pi' b)$

$\Pi' a$ = 647BA1C8 F922204C 5D0F4AD3 DD0E093B FF068454 D67B803C 82C0F082
876BC786 748E005C EBF2C2DA 67E2CA31 0CE29DED 0EFAF684 3C1660D0
549989A5 A2F46DF8

$\Pi' b$ = 68DC5B24 5DB19947 E3ABA4C5 0AE4C90F 4C2B5AD5 F55DDFAB 61D46A5E
7BFA66B3 32431664 E1268632 159F22B6 5F745302 5FFD9E93 1314D23A
69D6E89D 759CF6BF

R' = 06CD1062 D7B840D5 8B6973C5 7B075BF9 7877680F

F.13 IBS-2 机制

F.13.1 概述

在本示例中，F_p^2中的元素表示为$a + \sigma b$，其中a和b为F_p中的元素，σ为F_p^2中满足$\sigma^2 + 1 \equiv$

$0 \bmod p$的元素。将一个挠群中的点转换为另一个挠群中的元素（以实现配对的非退化性）的畸变映射定义如下：$\phi P: (P_X, P_Y) \rightarrow (\beta P_X, P_Y)$，其中$\beta \neq 1$为$F_p^2$中满足$\beta^3 - 1 \equiv 0 \bmod p$的元素。$<P, Q>$实现为两个输入点$P$和$\phi(Q)$上的约化 Tate 配对。

在本示例中，H_2实现为 SHA-1。H_1对给定消息M的实现为：$H_1(M) =$ I2P(BS2I(512, MGF1(M)))，其中 MGF1(M)为 SHA-1(M||0)||SHA-1(M||1)||SHA-1(M||2)||SHA-1(M||3)的最左侧 512 位。

F.13.2 示例 1：域F_p，512 位素数p，SHA-1

F.13.2.1 参数

域为F_p，其中p的十六进制表示如下：

p = B35FA5FD E47FA1AB BB1E57E9 3BA1FF96 38B89B99 5C49BE81 A38E3194
A0983816 4EE51FB9 1D285832 F9A05D63 9C8D9680 10C93A35 27E561F2
FD6A45CC 70ABA1FB

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + 1$。

q = 80000000 000FFFFF FFFFFFFF FFFFFFFF FFFFFFFF

β = (β_a, β_b)

β_a = 59AFD2FE F23FD0D5 DD8F2BF4 9DD0FFCB 1C5C4DCC AE24DF40 D1C718CA
504C1C0B 27728FDC 8E942C19 7CD02EB1 CE46CB40 08649D1A 93F2B0F9
7EB522E6 3855D0FD

β_b = 20C3F863 7D0F88AE 295531C6 82537070 5EFA0F99 39D1CDE4 F58A25F4
3722E229 5410CA05 B2360EA8 625C8203 626C5254 3E63404A B83448B0
3FDA177A 7A2DDB21

P = (P_X, P_Y)

P_X = 7EE6F118 9329ADB4 1E8CD405 2295F1A7 6096631D F065DD38 85FFF26B
8ED52022 7BFC3F3D 07FE9CFE 093424BA 9DBD4C0D 73FFF367 3CE2C922
F0B73C50 2992093B

P_Y = 67C855C9 E6B617BE B24B792A E9C3E21E 95F37006 25B91058 3BC0B293
C36A762C FEBA4266 038989CC 59797235 C6116D99 8A97B805 FF82C664
53720E5C 1DE95E3E

U = 50BD36F5 967F41F4 93B1F667 4AD462F6 2B7D69D8

V = $[U]P = (V_X, V_Y)$

V_X = 04863127 7E2F50F6 CCCD235D EC856E8C FB22F17A 232D6229 7B7ABA44
77639ADE F2CC4863 5C19ED50 9E7F3F2C C5EF1F5B 5225E45D 30C269A3
97978783 D7E0DD86

V_Y = 75651DE4 8D8AEC70 10B4084A 05EE2EFB 0D72CF41 4F5B6113 539198F5
DC2317AA 3B6DEE3E E800394F 470020FB E6EE9811 CDDC5991 4BD0CB38
2E203F01 0816E8B7

F.13.2.2 签名密钥与验证密钥

ID = “alice@network.com”

MGF1(ID) = 85499895 C3B4F412 56E84BDC E3D76405 33612345 D6BF0725 B7D0391
D53A680B 2DD96646 A32B244E DA614B5D 8E989340 473B7885 F243CE0

0FBD2661 43B85636

$\mathrm{I2P}(\mathrm{BS2I}(512, \mathrm{MGF1}(ID))) = (Y_X, Y_Y)$

Y_X = 8BB6EFE1 AF00D035 D0B286F1 94BC3128 A639949A B9D06405 95F837A
E131C3C6 B37F2905 4D58FB21 9FA190B7 A65BE61F 4A6945B5 4A1A36C
6166F047 2F63D303

Y_Y = 3D605915 84CF4C40 2E81145C 0047D9BE 2733A3C2 66F7A8F8 CA2F50C
635424C3 8AFF64FA F9B5CD04 4AD0236F BC7BEB72 0D1909DA CC3C3FE
3022CDB7 7946F37D

$X = [U]Y = (X_X, X_Y)$

X_X = B26A00EA 12877F1D 049DC4F2 4030CDB1 632AA9FA D48E209D CC1A116
A159CBDD 540F417D FC584FF9 4C959E64 3BFA2131 7EB54356 BB8650B
8803CB32 F607C87B

X_Y = 3527A3A6 E22290F9 0CC5ECBE F6CFE3DA F0F7EF16 327606C8 2456C9F
6C7A2818 84832B75 41D8EAAE DB1B0FDA 78335B84 C56FEF1F 88423C3
1393760B AE4520D1

F.13.2.3 逐消息数据

M = "abc"的 ASCII 形式 = 61 62 63

K = 1FF4E7C8 5257349D E625E51B 7374BF96 A732E142

$\Pi = (\Pi_X, \Pi_Y)$

Π_X = 509EF0AE 5E9161BF 37657765 213A5F0F 5BACAAF5 0D8487F1 71820AC4
79972203 06A8F304 9231377E 51402041 AC76953C A78AD3B5 AC37E9B3
34257AA7 66106EC5

Π_Y = 97A5D5ED 33F32DC1 D93FAF34 82AF186A CD947D5C DB3F900D AB720999
12789CD9 114AC35A C3887189 F74C36BF 0D3E8F9C 8F7AF48F 7FB2EB54
00DB8B69 1EC16528

$H = H_2(M \| \mathrm{FE2BS}(r, \Pi_X))$ = 223C3AA6 91EA9C1F 21065A74 C3FD2C0D 507ED8A6

F.13.2.4 签名

$R = \Pi = (\Pi_X, \Pi_Y)$

Π_X = 509EF0AE 5E9161BF 37657765 213A5F0F 5BACAAF5 0D8487F1 71820AC4
79972203 06A8F304 9231377E 51402041 AC76953C A78AD3B5 AC37E9B3
34257AA7 66106EC5

Π_Y = 97A5D5ED 33F32DC1 D93FAF34 82AF186A CD947D5C DB3F900D AB720999
12789CD9 114AC35A C3887189 F74C36BF 0D3E8F9C 8F7AF48F 7FB2EB54
00DB8B69 1EC16528

$S = (S_X, S_Y)$

S_X = 0FD05023 373BA8E8 26715427 1A474C0D B57E88CC 70EA10AA 445436ED
8DC59EF4 86958DC6 64240FC6 32742CED 9FC96B36 925C0D8E ECD0FD19
CAA7054C FCD34157

S_Y = 1B909401 7DFE3D51 C330C2F5 8CD5F77A D84D97C2 EE7B4A6C 3451596C
99C21809 D422B5D5 4309CA08 3DE45DA7 9F823282 64C15B87 1741795C

EDFAA568 D996A7E0

F.13.2.5 验证

$H = H_2(M \| \mathrm{FE2BS}(r, \Pi_X)) =$ 316A0C05 67D24048 1C653139 F7BEB3EE 4CA81252

$T_1 = \langle P, S \rangle = (T_1a, T_1b)$

$T_1a =$ 89E32524 8BFFAB75 8C90FF00 46CFE55B 9EAE6AB3 1D1A859E A5295A51
13E4C043 857D2535 ABD88412 AFD36F56 DDE8F61C 28DE8235 635B70DB
B29A34B5 C1151EA5

$T_1b =$ B32755EC D8547A25 ADED38D0 FB19A23E 6F8042D9 3F871D0F D6599872
EAF31160 0397AC76 4410285F 002AB197 73779C26 C0AF7080 64AE2322
2CA73C2B 3FFA2CC3

$T_2 = \langle V, \Pi + [H]Y \rangle = (T_2a, T_2b)$

$T_2a =$ 89E32524 8BFFAB75 8C90FF00 46CFE55B 9EAE6AB3 1D1A859E A5295A51
13E4C043 857D2535 ABD88412 AFD36F56 DDE8F61C 28DE8235 635B70DB
B29A34B5 C1151EA5

$T_2b =$ B32755EC D8547A25 ADED38D0 FB19A23E 6F8042D9 3F871D0F D6599872
EAF31160 0397AC76 4410285F 002AB197 73779C26 C0AF7080 64AE2322
2CA73C2B 3FFA2CC3

F.14 SM2 机制

F.14.1 示例 1：域F_p，256 位素数p，SM3

F.14.1.1 概述

本示例使用 SM3 作为哈希函数h。哈希码即为 SM3 的输出值。SM3 在 ISO/IEC 10118-3 中规定。

F.14.1.2 参数

域为F_p，其中p的十六进制表示如下：

$p =$ 8542D69E 4C044F18 E8B92435 BF6FF7DE 45728391 5C45517D
722EDB8B 08F1DFC3

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + aX + b$。

$a =$ 787968B4 FA32C3FD 2417842E 73BBFEFF 2F3C848B 6831D7E0
EC65228B 3937E498

$b =$ 63E4C6D3 B23B0C84 9CF84241 484BFE48 F61D59A5 B16BA06E
6E12D1DA 27C5249A

$G = (G_X, G_Y)$

$G_X =$ 421DEBD6 1B62EAB6 746434EB C3CC315E 32220B3B ADD50BDC
4C4E6C14 7FEDD43D

$G_Y =$ 0680512B CBB42C07 D47349D2 153B70C4 E5D7FDFC BFA36EA1
A85841B9 E46E09A2

$q =$ 8542D69E 4C044F18 E8B92435 BF6FF7DD 29772063 0485628D
5AE74EE7 C32E79B7

F.14.1.3 签名密钥与验证密钥

X = 128B2FA8 BD433C6C 068C8D80 3DFF7979 2A519A55 171B1B65
0C23661D 15897263

$Y = (Y_X, Y_Y)$

Y_X = 0AE4C779 8AA0F119 471BEE11 825BE462 02BB79E2 A5844495
E97C04FF 4DF2548A

Y_Y = 7C0240F8 8F1CD4E1 6352A73C 17B7F16F 07353E53 A176D684
A9FE0C6B B798E857

F.14.1.4 逐消息数据

M = “message digest”的 ASCII 形式

= 6D 65 73 73 61 67 65 20 64 69 67 65 73 74

ID = “ALICE123@YAHOO.COM”的 ASCII 形式

= 414C 49434531 32334059 41484F4F 2E434F4D

$ENTL$ = 0090

Z = F4A38489 E32B45B6 F876E3AC 2168CA39 2362DC8F 23459C1D
1146FC3D BFB7BC9A

$H = h(Z||M)$ = B524F552 CD82B8B0 28476E00 5C377FB1 9A87E6FC 682D48BB
5D42E3D9 B9EFFE76

K = 6CB28D99 385C175C 94F94E93 4817663F C176D925 DD72B727
260DBAAE 1FB2F96F

$\Pi = (\Pi_X, \Pi_Y)$

Π_X = 110FCDA5 7615705D 5E7B9324 AC4B856D 23E6D918 8B2AE477
59514657 CE25D112

Π_Y = 1C65D68A 4A08601D F24B431E 0CAB4EBE 084772B3 817E8581
1A8510B2 DF7ECA1A

F.14.1.5 签名

R = 40F1EC59 F793D9F4 9E09DCEF 49130D41 94F79FB1 EED2CAA5
5BACDB49 C4E755D1

S = 6FC6DAC3 2C5D5CF1 0C77DFB2 0F7C2EB6 67A45787 2FB09EC5
6327A67E C7DEEBE7

F.14.1.6 验证

$\Pi' = (\Pi'_X, \Pi'_Y)$

Π'_X = 110FCDA5 7615705D 5E7B9324 AC4B856D 23E6D918 8B2AE477
59514657 CE25D112

Π'_Y = 1C65D68A 4A08601D F24B431E 0CAB4EBE 084772B3 817E8581
1A8510B2 DF7ECA1A

R' = 40F1EC59 F793D9F4 9E09DCEF 49130D41 94F79FB1 EED2CAA5
5BACDB49 C4E755D1

F.14.2 示例 2：域F_{2^m}，$m = 257$，SM3

F.14.2.1 概述

本示例使用 SM3 作为哈希函数h。哈希码即为 SM3 的输出值。

F.14.2.2 参数

域F_{2^m}表示为对不可约多项式$x^{257} + x^{12} + 1$ 取模的多项式。

椭圆曲线定义为：在F_{2^m}上，$Y^2 + XY = X^3 + aX^2 + b$。

a = 0

b = 00 E78BCD09 746C2023 78A7E72B 12BCE002 66B9627E
CB0B5A25 367AD1AD 4CC6242B

G = (G_X, G_Y)

G_X = 00 CDB9CA7F 1E6B0441 F658343F 4B10297C 0EF9B649 1082400A
62E7A748 5735FADD

G_Y = 01 3DE74DA6 5951C4D7 6DC89220 D5F7777A 611B1C38 BAE260B1
75951DC8 060C2B3E

q = 7FFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF BC972CF7 E6B6F900
945B3C6A 0CF6161D

F.14.2.3 签名密钥与验证密钥

X = 771EF3DB FF5F1CDC 32B9C572 93047619 1998B2BF 7CB981D7
F5B39202 645F0931

Y = (Y_X, Y_Y)

Y_X = 01 65961645 281A8626 607B917F 657D7E93 82F1EA5C D931F40F
6627F357 542653B2

Y_Y = 01 68652213 0D590FB8 DE635D8F CA715CC6 BF3D05BE F3F75DA5
D5434544 48166612

F.14.2.4 逐消息数据

M = “message digest”的 ASCII 形式

= 6D 65 73 73 61 67 65 20 64 69 67 65 73 74

ID = “ALICE123@YAHOO.COM”的 ASCII 形式

= 414C 49434531 32334059 41484F4F 2E434F4D

$ENTL$ = 0090

Z = 26352AF8 2EC19F20 7BBC6F94 74E11E90 CE0F7DDA CE03B27F
801817E8 97A81FD5

H = $h(Z \| M)$

= AD673CBD A3114171 29A9EAA5 F9AB1AA1 633AD477 18A84DFD
46C17C6F A0AA3B12

K = 36CD79FC 8E24B735 7A8A7B4A 46D454C3 97703D64 98158C60
5399B341 ADA186D6

Π = (Π_X, Π_Y)

Π_X = 00 3FD87D69 47A15F94 25B32EDD 39381ADF D5E71CD4 BB357E3C
6A6E0397 EEA7CD66

Π_Y = 00 80771114 6D73951E 9EB373A6 58214054 B7B56D1D 50B4CD6E
B32ED387 A65AA6A2

F.14.2.5 签名

R = 6D3FBA26 EAB2A105 4F5D1983 32E33581 7C8AC453 ED26D339
1CD4439D 825BF25B

S = 3124C568 8D95F0A1 0252A9BE D033BEC8 4439DA38 4621B6D6
FAD77F94 B74A9556

F.14.2.6 验证

Π' = (Π'_X, Π'_Y)

Π'_X = 00 3FD87D69 47A15F94 25B32EDD 39381ADF D5E71CD4 BB357E3C
6A6E0397 EEA7CD66

Π'_Y = 00 80771114 6D73951E 9EB373A6 58214054 B7B56D1D 50B4CD6E
B32ED387 A65AA6A2

R' = 6D3FBA26 EAB2A105 4F5D1983 32E33581 7C8AC453 ED26D339
1CD4439D 825BF25B

F.15 中国 IBS 机制

F.15.1 概述

在本示例中，F_p^2中的元素A_0表示为$A_{0,1}\sigma + A_{0,0}$，其中$A_{0,0}$和$A_{0,1}$为F_p中的元素，σ为F_p^2中满足$\sigma^2 + 2 = 0 \bmod p$的元素。设ν为F_p^4中满足F_p^2内$\nu^2 - \sigma = 0$ 的元素，ω为F_p^{12}中满足F_p^4内$\omega^3 - \nu = 0$ 的元素。F_p^{12}中的元素表示为$A\omega^2 + B\omega + C$，其中A, B, C为F_p^4中的元素，后者分别表示为$A = A_1\nu + A_0$，$B = B_1\nu + B_0$，$C = C_1\nu + C_0$。F_p^{12}中的元素表示为向量$(A_{1,1}, A_{1,0}, A_{0,1}, A_{0,0}, B_{1,1}, B_{1,0}, B_{0,1}, B_{0,0}, C_{1,1}, C_{1,0}, C_{0,1}, C_{0,0})$，各分量属于$F_p$。$P$为 BN 曲线$y^2 = x^3 + 5$ 上的点，Q为对应六次扭曲线$y^2 = x^3 + 5\sigma$上的点。$\langle P, Q\rangle$实现为两个输入点P和Q上的最优 R-ate 配对，即：

$$\langle P, Q\rangle = [d(\phi(Q), P, u)\cdot f([u]\phi(Q), \pi_p(\phi(Q)), P)\cdot f([u]\phi(Q) + \pi_p(\phi(Q)), -\pi_{p^2}(\phi(Q)), P)]^{(p^{12}-1)/q},$$

其中，$p = 36t^4 + 36t^3 + 24t^2 + 6t + 1$，$q = 36t^4 + 36t^3 + 18t^2 + 6t + 1$，$u = 6t + 2$。$t$是 BN 曲线的参数，$\pi$是 Frobenius 自同态，满足$\pi_{p^i}(x, y) = (x^{p^i}, y^{p^i}), i = 1,2$。$\phi$是从 E'到$E$的群同态，定义为$E' \to E: (x, y) \mapsto (\sigma^{-\frac{1}{3}}x, \sigma^{-\frac{1}{2}}y)$。

有关最优 R-ate 配对实现的更多详细信息，可参见参考文献[44]。

在本示例中，h实现为 SM3。SM3 在 ISO/IEC 10118-3 中规定。

F.15.2 示例 1：域F_p，256 位素数p，SM3

F.15.2.1 参数

域为F_p，其中p的十六进制表示如下：

p = B6400000 02A3A6F1 D603AB4F F58EC745 21F2934B 1A7AEEDB E56F9B27
E351457D

椭圆曲线定义为：在F_p上，$Y^2 = X^3 + 5$。

q = B6400000 02A3A6F1 D603AB4F F58EC744 49F2934B 18EA8BEE E56EE19C
D69ECF25

P = (P_X, P_Y)

P_X = 93DE051D 62BF718F F5ED0704 487D01D6 E1E40869 09DC3280 E8C4E481 7C66DDDD

P_Y = 21FE8DDA 4F21E607 63106512 5C395BBC 1C1C00CB FA602435 0C464CD7 0A3EA616

Q = (Q_X, Q_Y)

Q_X = (85AEF3D0 78640C98 597B6027 B441A01F F1DD2C19 0F5E93C4 4806C11 D8806141, 37227552 92130B08 D2AAB97F D34EC120 EE265948 19C17AB F9B7213B AF82D65B)

Q_Y = (17509B09 2E845C12 66BA0D26 2CBEE6ED 0736A96F A347C8BD 56DC76B 84EBEB96, A7CF28D5 19BE3DA6 5F317015 3D278FF2 47EFBA98 71A0811 6215BBA5 C999A7C7)

U = 0130E7 8459D785 45CB54C5 87E02CF4 80CE0B66 340F319F 348A1D5B 1F2DC5F4

V = $[U]Q = (V_X, V_Y)$

V_X = (9F64080B 3084F733 E48AFF4B 41B56501 1CE0711C 5E392CFB AB1B679 1B94C408, 29DBA116 152D1F78 6CE843ED 24A3B573 414D2177 86A92DD 8F14D656 96EA5E32)

V_Y = (69850938 ABEA0112 B57329F4 47E3A0CB AD3E2FDB 1A77F335 89E1408 D0EF1C25, 41E00A53 DDA532DA 1A7CE027 B7A46F74 1006E85F CDFF073 0E75C05F B4E3216D)

F.15.2.2 签名密钥与验证密钥

hid = $0x^{01}$

ID = “Alice” = 416C6963 65

Y = $h_1(ID \| hid, q)$ = 2ACC468C 3926B0BD B2767E99 FF26E084 DE9CED8D BC7D5FBF 418027B6 67862FAB

X = $[U(U+Y)^{-1}]P = (X_X, X_Y)$

X_X = A5702F05 CF131530 5E2D6EB6 4B0DEB92 3DB1A0BC F0CAFF90 523AC875 4AA69820

X_Y = 78559A84 4411F982 5C109F5E E3F52D72 0DD01785 392A727B B1556952 B2B013D3

F.15.2.3 逐消息数据

M = “Chinese IBS standard”的 ASCII 形式

= 4368696E 65736520 49425320 7374616E 64617264

$\langle P, V \rangle$ = (4E378FB5 561CD066 8F906B73 1AC58FEE 25738EDF 09CADC7A 9C0ABC0 177AEA6D, 28B3404A 61908F5D 6198815C 99AF1990 C8AF3865 930058C 28C21BB5 39CE0000, 38BFFE40 A22D529A 0C66124B 2C308DAC 2299126 56F62B4F ACFCED40 8E02380F, A01F2C8B EE817696 09462C69 96AA923 FD863E20 9D3CE26D D889B55E 2E3873DB, 67E0E0C2 EED7A699 DCE28FE

9AA2EF56 83430786 0839677F 96685F2B 44D0911F, 5A1AE172 02EFD95
DF7338DB C577C66D 8D6C15E0 A0158C75 07228EFB 078F42A6, 604A3FC
FA9783E6 67CE9FCB 1062C2A5 C6685C31 6DDA62DE 0548BAA6 A30038B,
93634F44 FA13AF76 169F3CC8 FBEA880A DAFF8475 D5FD28A7 5DEB83C4
4362B439, B3129A75 D31D1719 4675A1BC 56947920 898FBF39 A5BF5D9
31CE6CBB 3340F66D, 4C744E69 C4A2E1C8 ED72F796 D151A17C 2325B94
3260FC46 0B9F73CB 57C9014B, 84B87422 330D7936 EABA1109 A5A7A71
81EE16F2 438B0AEB 2F38FD5F 7554E57A, AAB9F06A 4EEBA432 A7833DB
202E4E35 639D93FA 3305AF73 F0F071D7 D284FCFB)

K = 033C86 16B06704 813203DF D0096502 2ED15975 C662337A ED648835
DC4B1CBE

Π = (81377B8F DBC2839B 4FA2D0E0 F8AA6853 BBBE9E9C 4099608F 612C607
8ACD7563, 815AEBA2 17AD502D A0F48704 CC73CABB 3C06209B 87142E1
4CBD99E8 BCA1680F 30DADC5C D9E207AE E32209F6 C3CA3EC0 D800A1A4
2D33C731 53DED47C 70A39D2E, 8EAF5D17 9A1836B3 59A9D1D9 FC19F2E
FCDB8293 28620962 BD3FDF15 F2567F58, A543D256 09AE9439 0679194
ED30328B B33FD156 60BDE485 C6B79A7B 32B01398, 3F012DB0 BA59FE8
8DB88932 1CC2373D 4C0C35E8 4F7AB1FF 33679BCA 575D6765, F8624EB
435B838C CA77B2D0 347E65D5 E4696441 2A096F41 50D8C5ED 5440DDF,
0656FCB6 63D24731 E8029218 8A2471B8 B68AA993 89926849 9D23C897
55A1A897, 44643CEA D40F0965 F28E1CD2 895C3D11 8E4F65C9 0E3E741
B6DD52C0 EE2D25F5, 898D6084 8026B7EF B8FCC1B2 442ECF07 5F8A81C
EE99A624 8F294C82 C90D26BD, 6A814AAF 475F128A EF43A128 37F8015
4AE6CB92 CAD7D150 1BAE30F7 50B3A9BD, 1F96B08E 97997363 1131470
5BFB9A9D BB97F755 53EC90FB B2DDAE53 C8F68E42)

F.15.2.4 签名

$R = h_2(M_1 \| \Pi, q)$ = 823C4B21 E4BD2DFE 1ED92C60 6653E996 66856315 2FC33F55
D7BFBB9B D9705ADB

S = (S_X, S_Y)

S_X = 73BF9692 3CE58B6A D0E13E96 43A406D8 EB98417C 50EF1B29 CEF9ADB4
8B6D598C

S_Y = 856712F1 C2E0968A B7769F42 A99586AE D139D5B8 B3E15891 827CC2AC
ED9BAA05

F.15.2.5 验证

Π' = (81377B8F DBC2839B 4FA2D0E0 F8AA6853 BBBE9E9C 4099608F 612C607
8ACD7563, 815AEBA2 17AD502D A0F48704 CC73CABB 3C06209B 87142E1
4CBD99E8 BCA1680F, 30DADC5C D9E207AE E32209F6 C3CA3EC0 800A1A4
2D33C731 53DED47C 70A39D2E, 8EAF5D17 9A1836B3 59A9D1D9 FC19F2E
FCDB8293 28620962 BD3FDF15 F2567F58 A543D256 09AE9439 20679194
ED30328B B33FD156 60BDE485 C6B79A7B 32B01398, 3F012DB0 BA59FE8
8DB88932 1CC2373D 4C0C35E8 4F7AB1FF 33679BCA 575D6765, F8624EB
435B838C CA77B2D0 347E65D5 E4696441 2A096F41 50D8C5ED 5440DDF,

0656FCB6 63D24731 E8029218 8A2471B8 B68AA993 89926849 9D23C897
55A1A897, 44643CEA D40F0965 F28E1CD2 895C3D11 8E4F65C9 0E3E741
B6DD52C0 EE2D25F5, 898D6084 8026B7EF B8FCC1B2 442ECF07 5F8A81C
EE99A624 8F294C82 C90D26BD, 6A814AAF 475F128A EF43A128 37F8015
4AE6CB92 CAD7D150 1BAE30F7 50B3A9BD, 1F96B08E 97997363 1131470
5BFB9A9D BB97F755 53EC90FB B2DDAE53 C8F68E42)

R' = 823C4B21 E4BD2DFE 1ED92C60 6653E996 66856315 2FC33F55 D7BFBB9B
D9705ADB

附 录 G
（资料性）
签名方案比较

G.1 比较签名方案的符号和缩略语

为便于附录G的使用，采用以下符号和缩略语。

exp 模幂运算$(mod\ p)$

$exp-G_2$ G_2中的模幂运算

$GEXP$ 标量乘法

$GEXP-G_1$ G_1中的标量乘法

inv 求逆$(mod\ q)$

$pre-c$ 此操作可预先计算

$\langle\rangle$ 配对运算

G.2 签名方案比较

表 G.1 给出了基于Z_p^*的基于证书的机制的比较。

表 G.2 给出了使用椭圆曲线的基于证书的机制的比较。

表 G.3 给出了基于身份的机制的比较。

表 G.1 基于证书的机制（基于Z_p^*）比较

		DSA	KCDSA	Pointch*e*val/Vaudenay	SDSA
密钥长度	签名密钥	β位	β位	β位	β位
	验证密钥	α位	α位	α位	α位
签名长度		2β位	$\gamma+\beta$位	2β位	$\gamma+\beta$位
计算	密钥生成	1 次指数运算	1 次指数运算，1 次求逆	1 次指数运算	1 次指数运算
	签名生成	1 次指数运算，1 次求逆	1 次指数运算	1 次指数运算，1 次求逆	1 次指数运算
	签名验证	2 次指数运算，1 次求逆	2 次指数运算	2 次指数运算，1 次求逆	2 次指数运算

表 G.2 基于证书的机制（使用椭圆曲线）比较

		EC-DSA	EC-KCDSA	EC-GDSA	EC-RDSA	EC-SDSA	EC-FSDSA	SM2
密钥长度	签名密钥	β位	β位	β位	β位	β位	β位	β位
	验证密钥	2β位	2β位	2β位	2β位	2β位	2β位	2β位
签名长度		2β位	$\gamma+\beta$位	2β位	2β位	$\gamma+\beta$位	3β位	2β位
计算	密钥生成	1 次 GEXP	1 次 GEXP，1 次求逆	1 次 GEXP，1 次求逆	1 次 GEXP	1 次 GEXP	1 次 GEXP	1 次 GEXP，1 次求逆
	签名生成	1 次 GEXP，1 次求逆	1 次 GEXP	1 次 GEXP	1 次 GEXP	1 次 GEXP	1 次 GEXP	1 次 GEXP
	签名验证	2 次 GEXP，1 次求逆	2 次 GEXP	2 次 GEXP，1 次求逆	2 次 GEXP，1 次求逆	2 次 GEXP	2 次 GEXP	2 次 GEXP
a 未使用点压缩技术。								

注 1：严格来说，EC-FSDSA 签名的长度为 $2\beta' + \beta$，其中β'是β向上取整至 8 的倍数。这是 FE2BS 定义带来的产物，该函数始终输出长度为 8 比特整数倍的字符串。

注 2：GEXP 表示群元素指数运算，也称为标量乘法。

表 G.3 基于身份的机制比较

		IBS−1	IBS−2	中国 IBS
密钥长度	主私钥	β位	β位	β位
	主公钥	$\lceil\alpha/m\rceil$位	$\lceil\alpha/m\rceil$位	$\lceil\alpha/m\rceil$位
	签名密钥	$\lceil\alpha/m\rceil$位	$\lceil\alpha/m\rceil$位	$\lceil\alpha/m\rceil$位
	验证密钥	$\lceil\alpha/m\rceil$位	$\lceil\alpha/m\rceil$位	β位
签名长度		β+$\lceil\alpha/m\rceil$位	2*$\lceil\alpha/m\rceil$位	β+$\lceil\alpha/m\rceil$位
计算	密钥生成	1 GEXP-G_1	1 GEXP-G_1	1 次求逆，1 GEXP-G_1
	签名生成	1⟨⟩(pre−c)，1次 exp-G_2 ，1 GEXP-G_1	2 GEXP-G_1	1⟨⟩(pre−c)，1 次 exp-G_2 ，1 GEXP-G_1
	签名验证	1⟨⟩，1⟨⟩(pre−c)，1 次 exp-G_2	1 GEXP-G_1，2⟨⟩	1⟨⟩，1⟨⟩(pre−c)，1 次 exp-G_2 ，GEXP-G_1(pre−c)

附 录 H
（资料性）
选择机制的特征声明

表 H.1 提供了不同签名机制可能优势的声明。

表 H.1 机制声称的特性

签名机制	声称的特性
DSA	美国 FIPS，广泛接受
KCDSA	韩国标准，声称可证明安全[12][41]，签名生成和验证中无需模（$mod\ q$）求逆
Pointcheval/Vaudenay	DSA 变体，声称可证明安全[31]
SDSA	广为人知且经过研究，可证明安全[30][32][33]，高效，无需求逆
EC-DSA	美国 FIPS 和 ANSI 标准，声称可证明安全[13][14]，广泛接受，存储效率，带宽节省，计算效率
EC-KCDSA	韩国标准，声称可证明安全[12][41]，签名生成和验证中无需模（$mod\ q$）求逆，存储效率，带宽节省，计算效率
EC-GDSA	德国标准，签名生成中无需模 q 求逆，存储效率，带宽节省，计算效率
EC-RDSA	俄罗斯标准 GOST R 34.10-2012 和跨国标准 GOST 34.310-2004，声称可证明安全（无需随机预言机），存储效率，带宽节省
EC-SDSA	SDSA 的椭圆曲线版本，特性与 SDSA 类似
EC-FSDSA	Schnorr 变体，无需模（$mod\ q$）求逆，具有潜在的高效性
SM2	中国标准，声称可证明安全[45]，签名生成和验证中无需模（$mod\ q$）求逆，计算效率，存储效率，带宽节省，以及对强密钥替代攻击的可证明安全性[45]

参 考 文 献

[1] ISO/IEC 9594-8 | ITU-T Rec. X.509，信息技术 开放系统互连 目录：公钥和属性证书框架

[2] ISO/IEC 11770-3，信息技术 安全技术 密钥管理 第 3 部分：使用非对称技术的机制

[3] ISO/IEC 15945，信息技术 安全技术 支持数字签名应用的 TTP 服务规范

[4] 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：概述

[5] ISO/IEC 18032，信息技术 安全技术 素数生成

[6] ISO/IEC 18033-2，信息技术 安全技术 加密算法 第 2 部分：非对称密码

[7] 美国国家标准学会，金融服务业公钥密码：椭圆曲线数字签名算法（ECDSA），ANSI X9.62-2005，2005

[8] Barreto P., Kim H., Lynn B., Scott M. 基于配对的密码系统高效算法，CRYPTO 2002 会议论文集，LNCS 2442，第 354－369 页，Springer-Verlag，2002

[9] Bellare M., Garay J., Rabin T. 模幂和数字签名的快速批量验证，密码学进展——Eurocrypt 98 会议论文集，LNCS 第 1403 卷，第 236-250 页，Springer-Verlag，1998

[10] Bohli J.M., Rohrich S., Steinwandt R. 密钥替代攻击再探讨：考虑恶意签名者，Int. J. Inf. Secur. 2006，5，第 30－36 页

[11] Boneh D., & Franklin M. 基于 Weil 配对的基于身份加密，CRYPTO 2001 会议论文集，LNCS 2139，第 213-229 页，Springer-Verlag，2001

[12] Brickell E., Pointcheval D., Vaudenay S., Yung M. 基于离散对数签名方案的设计验证，PKC 2000 会议论文集，LNCS 1751，第 276-292 页，Springer-Verlag，2000

[13] Brown D.R.L. Generic Groups, Collision Resistance, and ECDSA. Des. Codes Cryptogr. ,35(1) pp. 119-152

[14] Brown D.R.L. 关于 ECDSA 的可证明安全性，载于《椭圆曲线密码学进展》（Blake I., Seroussi G., Smart N. 编），剑桥大学出版社，第二章，2005

[15] Cha J.C., & Cheon J.H. 基于间隙 Diffie-Hellman 群的基于身份签名，PKC 2002 会议论文集，LNCS 2567，第 18-30 页，Springer-Verlag，2002

[16] Erwin H., & Pascale S. 数字签名方案 EC-GDSA，德国联邦信息安全办公室，2005 年 12 月

[17] FIPS PUB 186-4，数字签名标准（DSS），美国国家标准与技术研究院，马里兰州盖瑟斯堡，2013

[18] Frey G., Müller M., Rück H. Tate 配对及其在椭圆曲线密码系统离散对数中的应用，IEEE Trans. Inf. Theory. 1999, 45 (5) 第 1717－1719 页

[19] Galbraith S. 超奇异曲线在密码学中的应用，Asiacrypt 2001 会议论文集，LNCS 2248，Springer-Verlag，2001，第 495－513 页

[20] Galbraith S., Harrison K., Soldera D. 实现 Tate 配对，ANTS-V 会议论文集，LNCS 2369，第 324－337 页，Springer-Verlag 出版社，2002 年

[21] GOST R 34.10-2012，俄罗斯联邦国家标准，“信息技术。加密数据安全。电子数字签名的签名与验证过程。”俄罗斯联邦标准化与计量委员会，2012 年。（俄文）

[22] Hess F. 基于配对的基于身份高效签名方案，SAC 2002 会议论文集，2002 年

[23] IEEE P1363，公钥密码学标准规范

[24] Koblitz N. 椭圆曲线密码系统。数学计算，1987 年，第 48 卷，第 203－209 页

[25] Lenstra A.K. 与 Verheul E.R. 选择加密密钥长度，载于《密码学杂志》，第 14 卷第 4 期，第 255-293 页，2001 年

[26] Menezes A. 椭圆曲线公钥密码系统。Kluwer 学术出版社，1993 年

[27] Menezes A.J.，Okamoto T.，Vanstone S. 将椭圆曲线对数归约到有限域中的对数。IEEE 信息论汇刊，1993 年，第 39 卷，第 1639－1646 页

[28] Miller V.S. 椭圆曲线在密码学中的应用，Crypto 1985 会议论文集，H.C. Williams 编，LNCS 218，第 417-426 页，柏林，Springer-Verlag 出版社，1986 年

[29] Miller V. Weil 配对及其高效计算，《密码学杂志》，第 17 卷第 4 期，第 235-261 页，2004 年

[30] Neven G.，Smart N.，Warinschi B. Schnorr 签名的杂凑函数要求。《数学密码学杂志》，2009 年，第 3 卷，第 69－87 页

[31] Pointcheval D. 与 Vaudenay S. 数字签名算法的可证明安全性，技术报告 LIENS-96-17，LIENS，1996 年

[32] Schnorr C.P. 智能卡的高效识别与签名。载于《密码学进展 Crypto’89》，LNCS 435，第 239-252 页，Springer-Verlag 出版社，1990 年

[33] Schnorr C.P. 智能卡的高效签名生成。《密码学杂志》，1991 年，第 4 卷，第 161－174 页

[34] Silverman R. 基于成本的对称与非对称密钥长度安全分析，RSA 实验室公告，第 13 卷，2000 年 4 月（2001 年 11 月修订）

[35] Silverman J.H. 椭圆曲线算术高级专题，GTM 151。Springer-Verlag 出版社，1994 年

[36] TTAK.KO-12.0001/R3，带附录的数字签名机制 第 2 部分：韩国基于证书的数字签名算法 KCDSA，2014 年。（韩文）

[37] TTAK.KO-12.0015/R2，带附录的数字签名机制 第 3 部分：使用椭圆曲线的韩国基于证书的数字签名算法 EC-KCDSA，2014 年。（韩文）

[38] Varnovskii N.P. 防篡改设备模型中数字签名的可证明安全性。《离散数学与应用》，2008 年，第 18 卷第 4 期，第 427－437 页

[39] Vaudenay S. DSS 上的隐藏碰撞，CRYPTO 1996 会议论文集，LNCS 1109，第 83-88 页，Springer-Verlag 出版社，1996 年

[40] Yen S. 与 Laih C. 适用于批量验证的改进数字签名。IEEE 计算机汇刊，1995 年，第 44 卷第 7 期，第 957－959 页

[41] Yum D.H. 与 Lee P.J. 随机预言模型下 KCDSA 的安全性证明。CISC 会议论文集，1999 年，第 173－180 页

[42] GM/T 0003.2，SM2 椭圆曲线公钥密码算法 第 2 部分：数字签名算法，2012 年（中文）

[43] GM/T 0044.2-2016，基于双线性配对的密码算法 第 2 部分：数字签名算法，2016 年（中文）

[44] Vercauteren F. 最优配对。载于《IEEE 信息论汇刊》，第 56 卷第 1 期：第 455-461 页，2010 年

[45] Zhang Z.F.，Yang K.，Zhang J.，Chen C. SM2 签名方案对抗广义密钥替换攻击的安全性

[46] 《安全标准化研究》，LNCS 第 9497 卷，第 140－153 页。Springer 国际出版公司，2015 年