ICS 35.040
L 80

中华人民共和国国家标准

GB/T 15843.5—2005/ISO/IEC 9798-5:1999

信息技术　安全技术　实体鉴别　第5部分:使用零知识技术的机制

Information technology—Security techniques—Entity authentication—Part 5:Mechanisms using zero knowledge techniques

(ISO/IEC 9798-5:1999,IDT)

2005-04-19 发布　　2005-10-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

GB/T 15843《信息技术　安全技术　实体鉴别》，由以下几部分组成：

——第1部分：概述

——第2部分：使用对称加密算法的机制

——第3部分：使用数字签名技术的机制

——第4部分：使用密码校验函数的机制

——第5部分：使用零知识技术的机制

本部分为GB/T 15843的第5部分，等同采用国际标准ISO/IEC 9798-5:1999《信息技术　安全技术　实体鉴别　第5部分：使用零知识技术的机制》(英文版)。

本部分的附录A，附录B，附录C，附录D，附录E和附录F是资料性附录。

本部分由中华人民共和国信息产业部提出；

本部分由全国信息安全标准化技术委员会归口；

本部分由中国电子技术标准化研究所、信息安全国家重点实验室起草。

本部分主要起草人：陈星、罗锋盈、胡磊、叶顶锋、张振峰、黄家英。

信息技术 安全技术 实体鉴别 第5部分:使用零知识技术的机制

1 范围

GB/T 15843的本部分详细说明了三种使用零知识技术的实体鉴别机制。所有在GB/T 15843的本部分中阐述的机制都提供单向鉴别。这些机制应用零知识的原理所构造,但是根据附录A的严格定义,对所有参数的选择,这些机制本身并不是零知识的。

第一种机制称为基于身份的机制。可信的认可机构为每一个声称者提供私有认可信息,该私有认可信息是作为声称者的标识数据和认可机构的私有密钥的函数计算出来的。

第二种机制称为基于使用离散对数的基于证书的机制。每一个声称者都拥有一对用于此机制的公开密钥和私有密钥对。每一个声称者身份的验证者必须拥有该声称者公开验证密钥的可信拷贝;其获取的方法已经超出了本标准的范围,但是它可以通过由可信第三方签名的证书的分发来获得。

第三种机制称为基于使用非对称加密系统的基于证书的机制。每一个声称者都拥有一对用于非对称加密系统的公开密钥和私有密钥对。每一个声称者身份的验证者必须拥有该声称者公开验证密钥的可信拷贝;其获取的方法已经超出了本标准的范围,但是可以通过由可信第三方签名的证书的分发来获得。

2 规范性引用文件

下列文件中的条款通过GB/T 15843的本部分的引用而成为本部分的条款。凡是注日期的引用文件,其随后所有的修改单(不包括勘误的内容)或修订版均不适用于本部分,然而,鼓励根据本部分达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件,其最新版本适用于本部分。

GB 15851—1995 信息技术 安全技术 带消息恢复的数字签名方案(idt ISO/IEC 9796:1991)

GB/T 15843.1—1999 信息技术 安全技术 实体鉴别 第1部分:概述(idt ISO/IEC 9798-1:1997)

GB/T 18238(所有部分) 信息技术 安全技术 散列函数(idt ISO/IEC 10118)

3 术语和定义

GB/T 15843.1—1999中确立的下列术语和定义适用于GB/T 15843的本部分。

3.1

非对称密码技术 asymmetric cryptographic technique

3.2

非对称加密系统 asymmetric encipherment system

3.3

非对称密钥对 asymmetric key pair

3.4

询问 challenge

3.5

声称者 claimant

3.6

解密 decipherment

3.7

可区分标识符 distinguishing identifier

3.8

加密 encipherment

3.9

实体鉴别 entity authentication

3.10

私有密钥 private key

3.11

公开密钥 public key

3.12

公开验证密钥 public verification key

3.13

随机数 random number

3.14

权标 token

3.15

可信第三方 trusted third party

3.16

单向鉴别 unilateral authentication

3.17

验证者 verifier

下列术语在 GB/T 18238.1 中定义。

3.18

散列函数 hash-function

将比特串映射到固定长度比特串的函数，它满足下列两条性质：

——为一个给定的输出找到能映射到该输出的一个输入在计算上是不可行的；

——为一个给定的输入找到能映射到同一个输出的另一个输入在计算上是不可行的。

另外 GB/T 15843 的本部分增加使用下列术语和定义。

3.19

认可机构 accreditation authority

实体群的所有成员所信任的实体，用于生成私有认可信息。

3.20

认可多重性参数 accreditation multiplicity parameter

一个正整数，它等于由认可机构为一个实体所提供的秘密认可信息项目的数目。

3.21

交换多重性参数 exchange multiplicity parameter

一个正整数，用以确定在一个鉴别机制的执行中应该进行多少次实体鉴别消息的交换。

3.22

标识数据 identification data

一个数据项目序列。包括分配给某一个实体的用于识别他的唯一标识符。

注：可能包括在标识数据中的数据项目包括：帐号、有效期限、序列号等等。

3.23

私有认可指数 private accreditation exponent

只有认可机构可知的数值，用于生成声称者的私有认可信息。这一数值应该保密，此数值与公开认可验证指数相关。

3.24

私有认可信息 private accreditation information

由认可机构向声称者提供的私有信息，也是声称者随后证明的知识内容，由此确立声称者的身份。

3.25

私有解密变换 private decipherment transformation

由非对称加密系统和非对称密钥对的私有密钥确定的解密变换。

3.26

公开认可验证指数 public accreditation verification exponent

一个实体群的所有成员一致认同的指数，它与模数一起确定私有认可指数的值。

3.27

公开加密变换 public encipherment transformation

由非对称加密系统和非对称密钥对的公开密钥确定的加密变换。

3.28

冗余身份 redundant identity

使用 GB 15851—1995 中所规定的技术，从一个实体的标识数据通过增加冗余而得到的数据项序列。

3.29

响应 response

声称者发送给验证者的数据项目，借助于它验证者可以核查声称者的身份。

3.30

证据 witness

向验证者提供声称者身份证据的数据项。

4 符号与标记

GB/T 15843.1—1999 中描述的符号和标记适用于 GB/T 15843 的本部分。

A 实体 A 的唯一标识符。

B 实体 B 的唯一标识符。

$Y \| Z$ 数据项 Y 和 Z 按照顺序连接的结果。

下述通用符号与标记要用到：

d 询问

D 响应

h 散列函数

r 随机数

$[x]$ 不大于 x 的最大整数。

mod 如果 i 是一个整数而 n 是一个正整数，那么 $i \bmod n$ 表示唯一的整数 j 满足：

a) $0 \leqslant j < n$，

b) $i-j$ 是 n 的整数倍。

在第 5 章基于身份机制一文中，要用到下列符号与标记：

$C_{A1}, C_{A2}, \cdots\cdots C_{Am}$ 实体 A 的私有认可信息。

gcd 两个整数的最大公因子，也即：gcd(a,b)表示同时为 a 和 b 的因子的最大正整数。

$I_{A1}, I_{A2}, \cdots\cdots, I_{Am}$ 实体 A 的标识数据。I_{Ai} 是实体 A 的标识数据的第 i 部分。

$J_{A1}, J_{A2}, \cdots\cdots, J_{Am}$ 实体 A 的冗余身份。J_{Ai} 是实体 A 冗余身份的第 i 部分。

k_S 由模数 n 确定的一个整数，它决定了一个实体的冗余身份的最大比特长度。

lcm 两个整数的最小公倍数，也即：lcm(a,b)表示 a 和 b 的公共倍数中的最小正整数。

m 认可多重性参数。

n 模数，等于素数 p 和 q 的乘积。

p 用于计算模数的素数

q 用于计算模数的素数

t 交换多重性参数

u 认可机构的私有认可指数

v 公开认可验证指数

W 证据

mod* 如果 i 是一个整数而 n 是一个正整数，那么 i mod* n 表示非负整数 j，它等于 i mod n 和 $n-i$ mod n 两个值中比较小的一个。如果 i 和 j 是两个整数并且 n 是一个正整数，那么 $i \equiv j \pmod{*n}$ 当且仅当 i mod* $n = j$ mod* n

$(a \mid n)$ 正整数 a 关于奇正整数 n 的 Jacobi 符号

注：设 p 为一个奇素数且 a 为一个正整数。a 关于 p 的 Legendre 符号，记作$(a \mid p)$，定义为：

$$(a \mid p) = a^{(p-1)/2} \bmod p$$

当 a 不是 p 的倍数时，$(a \mid p)$等于+1 或者−1，它取决于 a 是否等于一个整数模 p 的平方。素数 p 的倍数关于 p 的 Legendre 符号为零。

设 n 是一个奇的正整数满足 $n=pq$，其中 p 和 q 为素数，设 a 是一个正整数。那么 a 关于 n 的 Jacobi 符号，记作$(a \mid n)$，定义为：

$$(a \mid n) = (a \mid p)(a \mid q)$$

在不知道 n 的素因子分解的情况下，Jacobi 符号$(a \mid n)$可以有效地计算，参看[6]和[8]。

在第 6 章基于离散对数机制的上下文中，下列符号与标记被使用：

g——正整数，它是离散对数的基。

p——用作模的素数。

q——一个素数，是 $p-1$ 的因数。

yx——实体 X 的公开验证密钥。

zx——实体 X 的私有鉴别密钥。

在第 7 章基于可信公开变换机制的上下文中，使用下列符号与标记：

Px——实体 X 的公开加密变换。

Sx——实体 X 的私有解密变换。

5 基于身份的机制

本章阐述了一个实体鉴别机制，它使用基于身份的机制。

5.1 具体的要求

为了在一个实体群中使用这个机制，应该采取下列步骤：

a) 每一个希望充当声称者或者验证者的实体必须有产生随机数的方法；

b) 这个实体群应该指定一个认可机构。此认可机构应为这个群内所有成员所信任，以便保证其身份；

c) 需要选择一些参数，它们将决定该实体鉴别机制的操作。选取的参数应该以一种可信赖的方式让实体群的所有成员获知；

d) 每一个希望在鉴别机制中充当声称者的实体必须以某种方式获得标识数据。在此上下文中标识数据是一个长度由步骤 c) 中所选择的一个参数限定的比特串。依照约定的惯例对实体的标识是唯一的、有意义的；

e) 每一个希望在鉴别机制中充当声称者的实体应该由指定的认可机构签发私有认可信息；

f) 如果选择使用散列函数机制的版本，群内所有实体必须对于专用散列函数的使用达成一致（比如在 GB/T 18238 中描述的某一个散列函数）。

5.2 参数选择

参数应按下述方法进行选择：

a) 公开认可验证指数 v。某些具有实际优势的值，例如 2，3 和 $2^{16}+1=65\ 537$；

b) 模数 n。这个正整数应该由指定的认可机构选择。n 的值等于两个素数 p 和 q 的乘积。认可机构应该对 p 和 q 的值保密。素数 p 和 q 的选择方法应该使得任何实体在知道它们的乘积的情况下来推断它们是不可行的，这里的可行性是由鉴别机制的使用上下文所定义。

p 和 q 的值应满足下列条件：

——如果 v 是奇数，那么 $\gcd(p-1,v)=\gcd(q-1,v)=1$

——如果 v 是偶数，那么 $\gcd(p-1/2,v)=\gcd(q-1/2,v)=1$，且 $p-q$ 不是 8 的倍数。

模 n 的选取以下面的方式决定着另一个参数 k_s 的值：

$$k_s=[\log_2(n)]$$

换句话说，n 的二进制表示应包含 k_s+1 个比特。

n 的选择也决定着认可机构的私有认可指数（记为 u）的值，以这样的方式：值 u 应设置为最小的正整数以使得 $vu+1$ 是下述的倍数：

$\mathrm{lcm}(p-1,q-1)$ 若 v 是奇数，

$\mathrm{lcm}(p-1,q-1)/2$ 若 v 是偶数。

c) 认可多重性参数 m。这个正整数的选取应该与公开认可验证指数 v 和交换多重性参数 t 的选择相结合，它影响到方案的安全级别。

d) 交换多重性参数 t。这个正整数的选取应该与公开认可验证指数 v 和认可多重性参数 m 的选择相结合，它影响到方案的安全级别。

注 1：附录 B 中给出了这个机制的参数选择的指导。

注 2：当 $v=2$ 时这个机制就成为了 Fait-Shamir 方案，[3]。当 $v>2$，$m=1$ 且 v 为素数时，这个机制就成为了 Guillou-Quisquater 方案，[5]。

5.3 身份选择

在这个机制中，每个希望充当声称者的实体必须被分配标识数据，它是由 m 个部分构成的序列：$I_{A1}, I_{A2}, \cdots\cdots I_{Am}$。标识数据的每一部分最多包含 $8[(k_s+3)/16]$ 个比特。

实体鉴别机制将向验证者提供保证：该声称者确实是分配到这个标识数据的实体。

注 1：比如，这个标识数据序列可能这样构造：给实体分配一个唯一的识别比特串，然后依次把数字 1，2，……，m 的二进制表示附加到这个比特串的后面从而得到 $I_{A1}, I_{A2}, \ldots\ldots, I_{Am}$ 的值。在这个方法中，数字 1，2，3，……，m 的二进制表示可以方便地表示成具有相同长度的形式，如果需要就加以前缀 0。

注 2：如果一个实体的标识数据部分超出了所允许的最大长度，那么可以运用散列函数去处理标识数据的部件以获得 $I_{A1}, I_{A2}, \cdots\cdots, I_{Am}$ 的值。散列函数的例子可以在 GB/T 18238 中找到。

注 3：实体标识数据的过期可以通过在标识数据中包含有效日期来实现。实体标识数据的撤消可以通过在标识数据中包含序列号得到简化。

5.4 认可产生

为了产生一个实体 A 的私有认可信息，认可机构应该计算一列 m 个数字签名 $C_{A1}, C_{A2}, \cdots\cdots C_{Am}$。

更明确地说，对每一个 $i(1\leqslant i\leqslant m)$，$C_{Ai}$ 应按照下列程序进行计算。

a) J_{Ai}，A 的“冗余身份”的第 i 部分，它将使用 k_S 指定的值，依据 GB 15851—1995 中签名过程的前四步（‘填充’，‘扩展’，‘冗余’和‘截断与实现’），从 A 的标识数据的第 i 部分 I_{Ai} 计算而来。从这个过程中得到的值，表示为 IR，将按下述方法用于导出 J_{Ai}。

——如果 v 为奇数，那么 $J_{Ai}=IR$。

——如果 v 为偶数并且 $(IR \mid n)=+1$，那么 $J_{Ai}=IR$。

——如果 v 为偶数并且 $(IR \mid n)=-1$，那么 $J_{Ai}=IR/2$

b) C_{Ai} 由 J_{Ai} 按照下列公式计算得到：

$$C_{Ai}=(J_{Ai})^{u} \bmod n$$

提供给实体 A 的私有认可信息等于计算的签名 C_{A1}，C_{A2}，……C_{Am}。

对于每一个 $i(1\leqslant i\leqslant m)$，容易看出

$$(C_{Ai})^{v} J_{Ai} \equiv 1 \pmod n$$

5.5 鉴别交换

这个单向的鉴别机制涉及到声称者 A 和声称者 B 之间的下列信息交换，它能够使 B 核查 A 的身份。为了机制的正确运行，需要向 B 提供 A 所声称的标识数据，可以把它附加在机制中交换的某一个信息之后，或者采用其他的方法。

图 1 中说明了鉴别程序一个回合的过程。图中括弧内的数字对应于下面详细描述的交换步骤。

第一个权标（TokenAB_1）是由声称者发送给验证者的，其格式为：

$$\text{Token}AB_1=W$$

或者

$$\text{Token}AB_1=h(W \parallel \text{Text})$$

其中 W 为证据，h 是散列函数，Text 为可选择的文本字段。这个文本字段（可能为空）对于 GB/T 15843 本部分的范围之外的应用是可用的。参看 GB/T 15843.1—1999 附录 A 中文本字段的使用信息。如果这个文本字段是非空的那么 B 必须设法恢复出文本字段的值；这可能需要 A 随着 TokenAB_1 一起发送全部或者部分文本字段（可参看下面的注 1）。

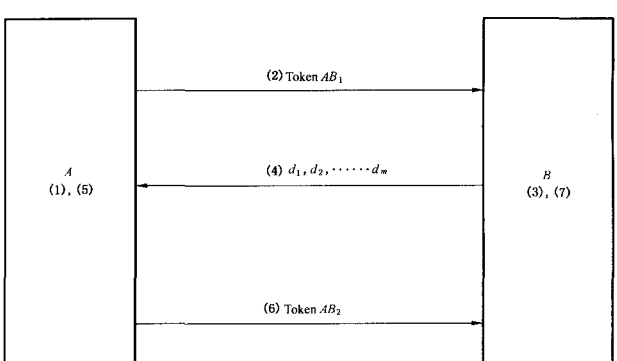

图 1 基于身份的机制

声称者发送给验证者的第二个权标(TokenAB_2)的格式为：

$$\text{Token}AB_2 = D$$

其中 D 为响应。

对于这个机制的每一次应用，下述鉴别程序应该执行 t 次(这里 t 为交换多重性参数)。只有当程序的 t 次执行都成功地完成，验证者 B 才接受声称者 A 的有效身份：

(1) 拥有私有认可信息 $C_{A1}, C_{A2}, \cdots\cdots C_{Am}$ 的实体 A 选择一个随机数 r，使得 r 为满足 $1 \leqslant r \leqslant n-1$ 的整数。这个整数由 A 保密。A 此时按照下列公式计算证据 W。

$$W = r^v \bmod n$$

(2) A 将 TokenAB_1 发送给 B。TokenAB_1 应等于 W 或者 $h(W \parallel \text{Text})$。

(3) 在收到 TokenAB_1 之后，B 选择一个随机整数序列 $d_1, d_2, \cdots\cdots d_m$，其中每一个 d_i 值应该在 0 到 $v-1$ 之间。这个整数序列就是询问。

(4) B 发送询问 $d_1, d_2, \cdots\cdots d_m$，给 A。

(5) 接收到询问 $d_1, d_2, \cdots\cdots d_m$ 之后，A 按下列公式由(秘密的)值 r 和私有认可信息 $C_{A1}, C_{A2}, \cdots\cdots C_{Am}$ 计算响应 D：

$$D = r \prod_{i=1}^{m} (C_{Ai})^{d_i} \bmod n$$

(6) A 发送 Token$AB_2 = D$ 给 B。

(7) 收到响应 D 之后，B 执行下列计算程序：

1) B 检查 D 是否满足 $0 < D < n/2$。如果不是那么 B 拒绝 A。

2) B 使用 5.4 步骤(a)所述的方法，根据 A 的标识数据 $I_{A1}, I_{A2}, \cdots\cdots, I_{Am}$ 计算出 A 的冗余身份 $J_{A1}, J_{A2}, \cdots\cdots, J_{Am}$。

3) B 按照下面的公式计算 W'的值：

$$W' = D^v \prod_{i=1}^{m} (J_{Ai})^{d_i} \bmod n$$

4) 如果在程序的第一次交换中发送的是 W，那么 B 核查他所计算的值 W'等于在程序的第一次交换中发送的 W 值。相应地，如果在程序的第一次交换中发送的是 $h(W \parallel \text{Text})$，那么 B 首先计算 $h(W' \parallel \text{Text})$，然后核查 $h(W' \parallel \text{Text})$ 是否等于在程序的第一次交换中发送的 $h(W \parallel \text{Text})$ 值。如果核查成功，那么机制在这一轮是成功的。否则，B 拒绝 A。

注 1：其他的信息可以随鉴别程序的任何轮的任何一次交换一起发送。特别需要说明的是，包含 A 的标识数据的信息可以和 TokenAB_1 一起在 t 轮鉴别过程的第一轮的首次交换(步骤(2))中发送。这些信息有助于 B 用来计算 A 的冗余身份和/或可选文本字段的值。

注 2：A 选择随机数 r 的过程要保证在认可信息的生命周期之内，这些选取的数值是独立的，这一点很重要。如果，例如，同一个值 r 使用两次，那么第三方就可能推导出 A 的部分或所有的私有认可信息，并因此成功地冒充 A。

注 3：关于 A 的冗余身份 $J_{A1}, J_{A2}, \cdots\cdots, J_{Am}$，$B$ 可以在任何阶段计算，也就是说，B 不需要等待收到了响应 D 之后才来计算 $J_{A1}, J_{A2}, \cdots\cdots, J_{Am}$。如果 B 使用这个机制频繁地验证 A 的身份，那么 B 可以存贮 $J_{A1}, J_{A2}, \cdots\cdots, J_{Am}$ 的值。

注 4：这个程序的 t 个回合可以并行执行，也就是说，在第一步 A 可以选择 t 个随机数 $r_1, r_2 \cdots\cdots r_t$，计算 t 个证据 $W_1, W_2, \cdots\cdots W_t$，把他们同时发送给 B，等等。如果采取“并行实现”，无论 t 值为多少，消息交换的总数将等于 3。

注 5：在程序的第一次交换中使用 $h(W \parallel \text{Text})$来代替 W 能够减少 TokenAB_1 的比特数，从而提高效率。

注 6：建议在这个机制中使用的私有认可信息仅仅用于鉴别的目的，而不要用于其他的应用中(例如数字签名的产生)。如果没有遵循这个建议，那么应该特别注意提防验证者利用声称者作为“签名谕示”；例如，这可以通过选取具有特殊的形式的询问而达到目的。

6 使用离散对数的基于证书的机制

在本章中详细阐述了一个使用离散对数的实体鉴别机制。

注：此机制称为 Schnorr 方案[9]。

6.1 具体的要求

为了在一个实体群中使用这个机制，需要采用下列步骤：

a) 每个希望充当声称者或者验证者的实体必须有产生随机数的方法。

b) 实体群中所有实体必须对三个正整数 p、q 和 g 达成一致。整数 p 必须选取为一个素数，同样 q 也必须选取为一个素数并且是 $p-1$ 的因子。最后 g 必须选取为在模 p 意义下阶为 q 的元素，即 g 必须满足：

1) $g^q \bmod p=1$

2) $g\neq 1$

值 p 和 q 的选择，应该使得给定任意的整数 $i(1<i<q)$，找到一个整数 j（如果存在）满足 $g^j \bmod p=i$ 在计算上是不可行的。

c) 群内所有实体必须对于专用散列函数的使用（例如在 GB/T 18238 中描述的散列函数之一）达成一致。

d) 每一个希望充当声称者的实体必须配备一个非对称密钥对，其选取如下所述。

e) 每一个希望充当验证者的实体，必须有一种方法获得将要验证身份的实体的公开验证密钥的可信拷贝。

注：为实体提供公开验证密钥的可信拷贝的具体方法超出了本标准的范围。例如这些可以通过使用公钥证书或者其他一些依赖于环境的方法得到。

6.2 密钥选择

在这个机制中，每一个希望充当声称者的实体 X 必须配备一个非对称密钥对（yx，zx）。其中 zx（私有密钥）是一个整数，其选取应满足 $0<zx<q$。相应的公开验证密钥 yx 应等于 $g^{zx} \bmod p$。

注：此机制的参数选择指导将在附录 B 中给出。

6.3 鉴别交换

这个单向的鉴别机制涉及到声称者 A 和验证者 B 之间的下列信息交换，它能够使 B 核查 A 的身份。

图 2 说明了这个鉴别机制。图中带括弧的数字对应于下面详细描述的交换步骤。

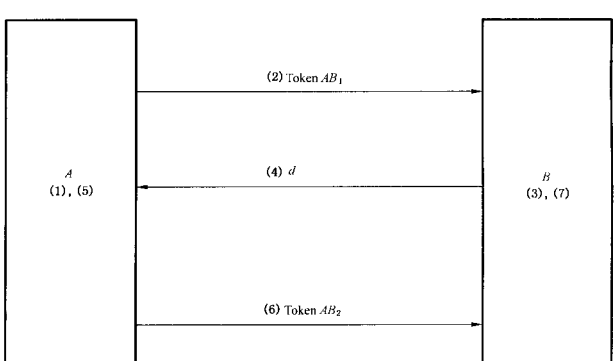

图 2 基于离散对数的机制

声称者发送给验证者的第一个权标(TokenAB_1)的格式为：

$$\text{Token}AB_1 = W$$

或者

$$\text{Token}AB_1 = h(W \parallel \text{Text})$$

其中 W 为证据，h 是散列函数，Text 为可选择的文本字段。这个文本字段(可能为空)对于 GB/T 15843 的本部分范围之外的应用是可用的。参看 GB/T 15843.1—1999 附录 A 中文本字段的使用信息。如果这个文本字段是非空的，那么 B 必须设法恢复文本字段的值；这可能需要 A 随同 TokenAB_1 一起发送文本字段的全部或部分(可参看下面的注 1)。

声称者发送给验证者的第二个权标(TokenAB_2)的格式为：

$$\text{Token}AB_2 = D$$

其中 D 为响应。

(1) 实体 A 选择一个随机数 r，使得 r 为满足 $1 \leqslant r \leqslant n-1$ 的整数。这个整数由 A 保密。A 此时按照下列公式计算证据 W

$$W = g^r \bmod p$$

(2) A 将 TokenAB_1 发送给 B。TokenAB_1 应等于 W 或 $h(W \parallel \text{Text})$。

(3) 收到 TokenAB_1 之后，B 应随机选择一个整数 d('询问')，这里 d 应满足 $0 \leqslant d < q$。

(4) B 发送询问 d 给 A。

(5) 接收到询问 d 后，A 按照下列公式由(秘密的)值 r 和 A 的私有密钥 z_A 计算出响应 D：

$$D = r - dz_A \bmod q$$

(6) A 发送 Token$AB_2 = D$ 给 B。

(7) 接收到响应 D 后，B 执行下列计算程序：

1) B 检查是否 $0 < D < q$。如果不是，那么 B 拒绝 A；

2) B 按照下列公式计算值 W'：

$$W' = (y_A)^d g^D \bmod p$$

3) 如果在程序的第一次交换中发送的是 W，那么 B 检验他计算的值 W' 是否等于在程序的第一次交换中发送的 W 值。相应地，如果在程序的第一次交换中发送的是 $h(W \parallel \text{Text})$，那么 B 首先计算 $h(W' \parallel \text{Text})$，然后核查 $h(W' \parallel \text{Text})$ 是否等于在程序的第一次交换中发送的 $h(W \parallel \text{Text})$ 值。如果 $h(W \parallel \text{Text}) \neq h(W' \parallel \text{Text})$ 那么这个机制失败并且 A 被拒绝。否则，B 接受 A。

注 1：其他的信息可以随鉴别程序的任何回合的任何一次交换一起发送。

注 2：A 选择随机数 r 的过程要保证在鉴别信息的生存周期之内，这些选取的数值是独立的，这一点很重要。如果，例如，同一个值 r 使用两次，那么第三方就可能推导出 A 的私有认可信息，并因此成功地冒充 A。

注 3：建议在这个机制中使用的密钥对仅仅用于鉴别的目的，而不要用于其他的任何应用中(例如数字签名的产生)。如果没有遵循这个建议，那么应该特别注意以防止验证者利用声称者作为"签名谕示"；例如，这可以通过要求询问具有特殊的选取形式而达到目的。

注 4：在程序的第一次交换中使用 $h(W \parallel \text{Text})$ 来代替 W 能够减少 TokenAB_1 的比特数，从而提高效率。

7 使用非对称加密系统的基于证书的机制

本章中详细说明了一个使用非对称加密系统的实体鉴别机制。

注：此机制源自 Brand-Damgard-Landrock-Pedersen 方案[1,8]。

7.1 具体的要求

为了在一群实体中使用此机制，需要采取下列步骤。

a) 每一个希望充当验证者的实体必须有产生随机数的方法。

b) 实体群中所有实体必须在两个密码函数的使用上达成一致：非对称加密系统、散列函数(例如

在 GB/T 18238 中描述的其中一个散列函数)。

c) 每一个希望充当声称者的实体必须配备一对用于非对称加密系统的非对称密钥对。

d) 每一个希望成为验证者的实体必须有一种方式获得将要验证身份的实体的公开验证密钥的可信拷贝。

注：为实体提供公开验证密钥的可信拷贝的具体方法超出了本标准的范围。这些方法可以从通过使用诸如公钥证书或其他一些依赖于环境的方法得到。

7.2 鉴别交换

这个单向的鉴别机制涉及到声称者 A 和验证者 B 之间的下列信息交换，它使得 B 能够核查 A 的身份。

图 3 说明了这个鉴别机制。图中带括弧的数字对应于符合下面详细描述的交换步骤。

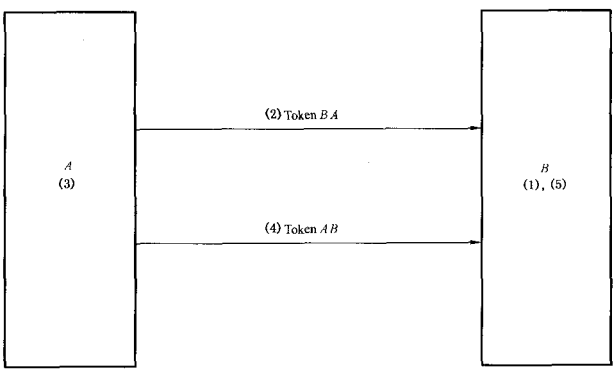

图 3 基于可信公开变换的机制

由验证者发送给声称者的权标(TokenBA)的格式为：

$$\text{Token}BA=d$$

其中 d 为询问。由声称者发送给验证者的权标(TokenAB)的格式为：

$$\text{Token}AB=D$$

其中 D 为响应。

(1) 实体 B 选择一个随机数 r，这个数由 B 秘密保存。B 接下来计算 $h(r)$。这个随机数 r 的选择应该使得 $r \parallel h(r)$ 位于 A 的公开加密变换 P_A 的定义域之内。B 然后按照下面的公式计算询问 d：

$$d=P_A(r \parallel h(r))$$

(2) B 将 Token$BA=d$ 发送给 A。

(3) 收到 TokenBA 之后，A 执行下列计算步骤：

1) A 通过计算 $r \parallel h(r)=S_A(d)$ 恢复 r 值。这里 S_A 为 A 的私有解密变换。

2) A 利用恢复的值 r 重新计算 $h(r)$，如果这个值不等于从 TokenBA 得来的值，那么 A 终止这个机制。如果计算出的 $h(r)$ 值与从 TokenBA 恢复出的值相同，那么 A 置 $D=r$。

(4) A 发送 Token$AB=D$ 给 B。

(5) 接收到 TokenAB 之后，B 比较 D 和 r。如果 $r \neq D$，那么机制失败，并且 A 被拒绝。如果 $r=D$，那么 B 接受 A。

注 1：其他的信息可以随同这个机制的交换一起发送。

注 2：B 选择随机数 r 的过程要保证：在 A 的非对称密钥对的生命周期之内同一个数值选取两次的概率趋近于零，

这一点很重要。如果，同一个数 r 使用两次，那么在它第一次发送的时候，截获到该响应 D 的第三方，在 B 第二次发送数值 r 后，通过重放 D 作为对 B 的响应就能够假扮 A 来欺骗 B。不过，重新使用以前有效的 r 值只会使这个机制应用的一个特殊回合失效。

注 3：建议在这个机制中使用的密钥对仅仅用于鉴别的目的，而不要用于任何其他的应用中（例如消息加密）。如果没有遵循这个建议，那么应该特别注意以防止验证者利用声称者作为“解密谕示”；例如，这可以要求散列值 $h(r)$ 具有特殊的格式而达到目的。

附 录 A
（资料性附录）
零知识机制的原理

A.1 简介

在非对称密码技术的使用过程中，鉴别交换的潜在弱点在于验证者可能滥用机制而危及声称者的私有密钥。当使用非对称密码的时候，声称者使用他的非对称密钥对的私有密钥来计算验证者询问的响应。验证者可能通过巧妙的选择询问来获得关于声称者私有密钥的信息，而这些信息是不可能仅仅从声称者的公开密钥的知识中得到的。

这种滥用密码消息交换的方式就是众所周知的把声称者作为“谕示”使用的方式，因为声称者按照验证者的要求提供有关其私有密钥的信息。零知识鉴别机制的思想就是通过谨慎的消息设计来消除这个潜在的威胁。这是通过确保验证者不能把声称者用作为“谕示”而实现的。

A.2 零知识机制的需要

在涉及现代计算机网络的应用当中，对安全服务的需求诸如用户鉴别、抗抵赖等等，正在被广泛地认可并不断地增长。一个用户为了能够使用这样的服务，他必需拥有该用户特定的私有信息。例如口令、数字签名系统的私有密钥，等等。

当然，对系统的安全来说，保持私有信息的秘密性是必须遵循的，也就是说不泄露给其他潜在的敌方。另一方面，私有信息必须用作软件或硬件模块的输入来计算和发送代表用户的消息。如果这些信息没有恰当地使用，私有信息的私秘性就可能受到损害，甚至被完全毁坏。一个明显的例子是用户通过明文形式发送口令来让主机识别自己。这就完全暴露了用户的私有信息，会直接导致任何在线窃听的人都能够冒充所有口令被拦截的用户。

这个例子中有过多的信息被传递。为了说明这一点，注意到就主机的角度而言，只有两种可能性：用户拥有正确的口令或者没有正确口令。在信息论中，这意味着真正需要传递的信息仅有 1 个比特。因此，通过发送完整口令的方法，我们就传递了比实际需要的多得多的信息，这就是窃听这些实际问题的理论背景。

我们自然会问：能不能设计出一些利用私有信息的协议？它们所传递的信息量恰好是我们打算传递的，一点儿也不多余。通俗地讲，这正好是零知识机制所具有的性质。例如考虑这样一个情形：在非对称密码系统中，用户 A 分配到了密钥对（P_A，S_A），其中 P_A 是 A 的公开密钥，而 S_A 是 A 的私有密钥。然后利用零知识机制，A 能够使 B 相信 A 拥有对应于 P_A 的私有密钥，而不泄露除此任何事实之外的任何信息。因为 A 是唯一有权使用 S_A 的用户，这个协议能够用于鉴别。在此方案中，零知识的特性保证了 B 不会获得任何可以帮助他以后非法冒充 A 的东西。

零知识的性质是通过设计一个可以由验证者单独模拟的会话来实现的。这就直观地证明了验证者不会从声称者那里得到有关声称者私有密钥特性的任何信息，而这些信息是验证者自己不能从相应的公开密钥中获取的。这也意味着一个观察者从构成这个机制的消息交换中不能判断出声称者是否真正参与了交换，或者这个交换只是由验证者模拟的。

零知识机制本身的特性要求使用非对称密码技术。在零知识机制的严格定义下，实际上是没有可能去实现的。事实上，在 GB/T 15843 的本部分中所阐述的机制的更好的描述应该是“秘密保护机制”。然而，零知识机制的概念是已经建立起来的众所周知的密码学理论的一部分。为此，我们在这里使用了这个术语。

A.3 定义

再接近正式定义一些，零知识机制发生在两个参与方之间：声称者 A 和验证者 B。声称者尽力使验证者相信某一个陈述是真实的。例如，这个陈述可能是“我知道对应于 P_A 的私有密钥”。为了使 B 确信，声称者和验证者首先交换一些消息，然后 B 决定接受或拒绝声称者的证明。

这样一个机制需要三个本质上的特性：

——完备性。如果 A 的陈述是真实的，那么 B 应该以极大的概率接受它。

——合理性。如果 A 的陈述不是真实的，那么无论 A 怎么做，B 应该以极大的概率拒绝它。

——零知识。无论 B 是如何做的，B 所接收到的只是 A 的陈述为真。更确切一点说：无论在与诚实的声称者会话时候 B 接收到了什么，B 应该同完全不与 A 会话一样容易由自己计算出来。这意味着 B 自身能够模拟会话，所产生的会话看上去好象的确是在与 A 会话一样。

A.4 一个例子

考虑下面的例子，它是 Fiat-Shamir 机制[3]的一个简化方案。这里，我们给定了一个模数 n 和一个关于模 n 的数，称作 y。在这个方案中，A 的陈述是“我知道 y 关于模 n 的一个平方根”。注意到 x 是 y 关于模 n 的一个平方根，当且仅当 $x^2 \bmod n = y$。

A 和 B 之间的会话将如下进行：

重复 t 次：

1. A 随机选择 r（这里 $2 \leqslant r \leqslant n-1$），将其模 n 平方并发送给 B。
2. B 随机选择比特 b 等于 0 或 1，发送给 A。
3. 如果 b 等于 0，那么 A 发送

$$z = r$$

给 B，否则 A 发送

$$z = rx \bmod n,$$

这里 x 是 y 关于模 n 的平方根，这是 A 所知道的。

4. B 首先检查 $z \neq 0$；如果 $z=0$ 那么 B 拒绝 A 并且终止程序。如果 $b=0$，B 就检查

$$z^2 \equiv r^2 \pmod n。$$

相应的，如果 b 等于 1 那么 B 检查

$$z^2 \equiv r^2 y \pmod n$$

如果检查是正确的就继续，否则 B 拒绝并且终止程序。

不难看出，如果 A 和 B 都遵循上述程序，那么 B 将不会拒绝 A；将 z 平方意味着把 r 或 rx 平方，那将给出结果 $r^2 \bmod n$ 或 $(rx)^2 = r^2 x^2 = r^2 y \bmod n$。

另一方面，如果在 t 次迭代的任何一次中，A 对于 $b=0$ 和 $b=1$ 都能够提供正确的答案，这就意味着 A 能够提供 z_0 和 z_1，使得

$$z_0^2 = r^2 \bmod n$$

并且

$$z_1^2 = r^2 y \bmod n。$$

通过将第一个方程式插入到第二个，可以直接地看出来数 $z_1/z_0 \bmod n$ 是 y 的一个平方根（$z_0 \neq 0$ 并且 z_0 必须以极大的概率和 n 互素）。如果 A 能够计算出具有在这样性质的 z_1 和 z_0，那么他就能够计算 z_1/z_0，因此关于他知道 y 的一个平方根的陈述就是真实的。但是相反地，如果 A 是在欺骗，他并不知道 y 的平方根，在 t 次迭代的每一次中他必定不能正确回答 b 的至少一个值。因此一个欺骗的声称者使验证者确信的概率至多为 2^{-t}。比如重复 20 次，我们就把这个概率缩小到约百万分之一。从而也满足了合理性的要求。至于零知识，注意到验证者在会话结束之后所留下的是两个数 z 和 r^2，满足

$$z^2 \equiv r^2 \pmod{n}$$

或者

$$z^2 \equiv r^2 y \pmod{n}$$

但是这些事情实际上验证者不用与 A 会话，自己就可以生成。B 只要选择一个随机的 z，定义

$$r^2 = z^2 \bmod n$$

或者

$$r^2 = z^2 / y \bmod n。$$

假如这样的话，这种计算 r^2 和 z 的方式不同于声称者的计算方式，但是这个事实是无关紧要的；二者的计算结果的分布是完全一样的，也就是说，不可能分辨其差异。因此，除了得知 A 知道 y 的一个平方根这个事实之外，B 不能得到任何他自己不能计算的信息。

让我们在这里预先考虑一个经常会提问到的问题。如果验证者不掌握 y 的一个根就能够自己生成很好的会话，那么为什么当声称者产生一个类似会话的时候，验证者就应该确信他的话呢？答案是当 B 模拟这个协议的时候，他可以自由地以“相反的方向”产生数字，也就是说，他首先选择 z 然后寻找符合条件的 r^2。在协议的实际执行中，A 没有这个机会。验证者期望在选择 b 之前看到 r^2，然后声称者必须找到一个正确的 z。

尽管我们在这里已经解释了几个技术难点，但存在讨论的要点：为什么机制具有零知识的特性。

A.5 基本的设计原理

前一部分的例子中，涉及到了支撑几乎所有已知的零知识机制的两个基本设计思想之一，也就是：

声称者发送一个“证据”给验证者。然后 B 从某一组问题之中询问 A 一个问题。如果 A 是欺骗的，他就不能够回答所有可能的问题。因此我们就有机会抓住他。另一方面 A 的回答从没多于一个问题，而这仅有的一个回答对验证者来说无法揭示出任何东西。这个设计思想构成了在第 5 和第 6 章中所阐述的机制的基本原理。

另外一个设计思想，也是构成第 7 章中所阐述的机制的基本原理，基于下述说明：

验证者向声称者询问一个问题，而验证者已经知道了这个问题的答案。协议必须确保事实就是这样的。如果 A 是诚实的，他能够容易地计算出正确的答案；但是如果他是欺骗的，他不会比随机地猜测做得更好，大多数时候的回答是不正确的。

另一方面，当 B 接收到答案时，他已经知道 A 将要说的话了，从而这个机制具备了零知识的特性。

一个简单的例子是在一个公开密钥系统中，当 A 必须证明他拥有一个私有密钥的时候，验证者能够利用 A 的公开密钥加密一个随机消息，要求 A 返回解密的消息。只有知道正确的私有密钥的用户才能做到这一点。为了得到零知识性，我们必须确保 B 真正预先知道这个消息。本标准包含了一个实现这个方法的例子，也就是要求 B 展示关于这个消息的某些信息（证据）。

准确理解零知识协议的正式定义在[2]和[4]中给出。

附 录 B
（资料性附录）
参数选择指南

本附录对于 GB/T 15843 的本部分中定义的两个机制的参数选取提供指导。

B.1 基于身份的机制的参数选择

本部分内容对参数 m，n，p，q，v 和 t 的选取进行指导。GB/T 15843 的本部分 5.2 中的注 2 也与这些参数的选择有关。

a) 模数 n。如 5.2 所述，素数 p 和 q 应按这样的方法来选择：任何实体知道了他们的乘积 n 来推导他们都是不可行的，这里的可行性根据鉴别机制使用的上下文来定义。

b) 公开认可验证指数 v。某些小素数 v 值，例如 2，3 或 $2^{16}+1$（对于后者有代表性地结合 $t=1$ 或 2）对于降低计算证据 W 的计算复杂性有实际的优势；一般来说，选取相对小的 v 值将降低声称者的计算复杂性。

c) 多重性参数 m，t。v^{-m} 的值等于一个不诚实的声称者在协议的每一次重复中，通过事先“猜测”询问 d_1，d_2……，d_m 的值而进行冒充攻击的成功概率。因此 m 和 t 的选择应该使得 v^{-m} 的值小于一个概率值门限，该值取决于应用的敏感度。对于大多数应用来说，2^{-16} 到 2^{-40} 之间的值是适当的，这里确切的值的选择将取决于风险评估。

结合这个机制使用的标识数据的形式需要小心地选择，尤其是如果标识数据需要撤销或终止，对应于标识数据的私有认可信息就应该兼顾到。更为明确地说，一个实体标识数据的过期可以通过在标识数据中包含有效日期来实现。可以在鉴别程序 t 次迭代之第一个回合的首次交换中包含有效日期信息，从而让验证者得到这个过期信息。

用类似的方法，通过在标识数据中包含序列号，可以实现一个实体的标识数据的撤销。验证者可以在标识数据的吊销“黑名单”列表中检查这个序列号。

B.2 使用离散对数的基于证书的机制的参数选择

本部分内容对参数 d，g，p 和 q 的选择进行指导。

a) 素数 p，q 和基 g。如 6.1 所述，p 和 g 的选取应该使得：给定任意一个整数 $i(1<i<q)$，找到一个整数 j（如果存在）满足 $g^j \bmod p=i$ 在计算上是不可行的。这样的整数 j 就是通常所说的 i 对于基 g 关于模 p 的离散对数。

计算上的可行性由鉴别机制使用的上下文定义。素数 p 和 q 的比特长度提供了计算离散对数的复杂度下界，因此 p 和 q 的长度必须小心地选择。

素数 p 的选择可以使得 q 的二进制表示的拷贝嵌入到 p 的二进制表示中。这样选择 p 和 q 的方法在存储空间和/或通信带宽非常珍贵的情况下是有用的。C.2.1.1 提供了这样的 p，q 对的例子。

若存在一个小于 q 整除 $p-1$ 的奇数因子，那么[7]中所述类型的攻击可能危及用户密钥的安全性。为了防止这样的攻击，p 和 q 的选择应该使得 $(p-1)/2q$ 没有小于 q 的素因子；理想地，$(p-1)/2q$ 应该为素数。

b) 询问 d。假设 d 在范围 $0 \leqslant d \leqslant 2^D-1$ 内随机选择，其中 D 是某一个正整数。值 2^{-D} 等于一个不诚实的声称者通过“猜测”询问 d 的值而进行冒充攻击的成功概率。因此 D（d 的比特长度）的选择应该使得 2^{-D} 小于由应用的敏感度而确定的概率值门限。对于大多数应用来说，2^{-16} 到 2^{-40} 之间的值是适当的，也就是说：D 的值在 16 到 40 之间，这里确切值的选择将取决于风险评估。

附 录 C
（资料性附录）
实 例

本附录给出 GB/T 15843 的本部分所阐述的实体鉴别机制的计算例子。这个附录的所有整数都以十六进制表示法给出。

这里给出的例子仅仅是为了说明和帮助验证这些机制的具体实现，在实际中不应该用。

C.1 基于身份的机制

C.1.1 公开指数为 2 的例子

C.1.1.1 参数选择

在这个例子中公开验证指数 v 为 2，一个偶数。因此秘密的素因数 p 和 q 必须满足：

$$\gcd(\frac{p-1}{2}, v) = \gcd(\frac{q-1}{2}, v) = 1$$

且，

$p-q$ 不是 8 的倍数。

p = f859 cdc6 f78f d206 a8d2 e78c bfc8 2735 5798 5d16 cbf9 431f abfc c16f 9ca9 3a5e f099 d3e8 3fe0 c67e 31f5 77dd ccf1 8287.

q = fef3 6abf 2aaf afa7 1c0b ca24 efe2 fb28 3366 1fb9 266f 9046 3c78 aa54 4a7c e2d8 9e56 071e 42db 00b3 c87e dc89 563a 02fb.

公开模数 n 为 768 比特长。

n = f755 3ef8 611b c569 0a2e 4d13 801a 94be 4dc8 fe2d da6c 6e11 586e 1941 81fb 96bf dc09 4d04 edbe ed1d 22ce 1fae 689b a233 3298 7fd7 9ef8 715f 1f5a 5eb4 b41f 45ea fcc5 4f32 5f21 5135 2930 8ff9 d8cd 5738 3801 fce9 7b51 f50f 8192 b0e1 c066 085d.

k_S=767

认可机构的私有认可指数 u 为最小的正整数满足：$uv+1$ 是 $\frac{\mathrm{lcm}(p-1, q-1)}{2}$ 的倍数。

u = leea a7df 0c23 78ad 2145 c0a2 7003 5297 c9b9 1fc5 bb4d 8dc2 2b0d c328 303f 72d7 fb81 29a0 9db7 dda3 a459 c3f5 cd13 7446 2769 68ea 2f97 1df6 2b4f 75a0 608e 8471 ae38 da4c 4d97 0fb9 e817 6486 be34 e740 1552 443c 5f12 c5bb b0e3 cb8f 53a7 505b.

m=8，t=3

C.1.1.2 身份选择

标识数据由 m=8 部分的序列组成。这些身份数据使用串'AlexAmple'并以一个 16 比特字段数字为后缀构成。

I_{A1} = 416c 6578 2041 6d70 6c65 0001

I_{A2} = 416c 6578 2041 6d70 6c65 0002

I_{A3} = 416c 6578 2041 6d70 6c65 0003

I_{A4} = 416c 6578 2041 6d70 6c65 0004

I_{A5} = 416c 6578 2041 6d70 6c65 0005

I_{A6} = 416c 6578 2041 6d70 6c65 0006

I_{A7} = 416c 6578 2041 6d70 6c65 0007

I_{A8} = 416c 6578 2041 6d70 6c65 0008

C.1.1.3 认可产生

J_{A1} = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2456 ee00 e301 9301 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e316.

C_{A1} = 79b7 7f76 b264 a2e0 bc4c e8f9 f29a 2175 99b4 2567 6dda 9360 228b ede5 748a d735 b2e9 bcf8 de99 6c8a 87db f920 26f4 b81e f97f 2b18 e50c 526b 2a40 f619 7d72 d7da 7d2e a641 2c2a fd97 df62 dfc4 56eb b043 8a99 1880 8749 387a 4a52 4a78 5049 4be6.

J_{A2} = 29a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7281 49a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7281 48a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7293.

C_{A2} = 41fb 8c2e c141 60fc 896b 1f36 d68a 4f8e 7a31 1226 31e2 28ea 568e c98e b09a 0e88 3500 21c1 8ac6 f81a 9f29 e8d2 25b0 8795 40b8 1791 e0ff 0cab 4aca 6e7c e17d c59c bc7e c931 9d92 beb5 8433 111e 14fd f601 6494 536f 2bc9 c692 a1f0 1da5 bd5d 8b90.

J_{A3} = 03c1 c485 28ac 5b9b 639a 0123 c093 6f2e d642 89e2 799a a434 2377 92d3 3ef2 d055 2cd8 3cfc 68ea 6a70 e310 1e72 5724 9c92 48e9 fd19 7ff1 d126 4c62 2ba8 8cac f99f daed 0adc e206 7e29 9dd9 0a15 3868 5d3b 396a af56 0332 fc84 46d0 2ab5 69fa 6cc6.

J_{A4} = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e904 9341 2a6d fe70 276c 2465 ee00 e904 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e904 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e946.

C_{A4} = 0e78 f7fe d61e 0934 ce1d 5c30 e3d8 7e50 def0 f339 06ff cadb ac33 fced 95c4 8579 d651 33fe 1bd6 963a 8a1e 56c5 d318 a94c fdab 5c27 9a61 25c5 07ed d1da 4aec a673 47cc 78f6 6a2b a671 e432 5d94 3b18 ad1e a2f2 f51e 2301 5070 ede3 fd92 1ea2.

J_{A5} = 29a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7202 c9a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7202 c8a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 722b.

C_{A5} = 2be9 2edd 3b6c 4977 ffe7 3b6f d0c9 1835 1b04 2b0a 33b9 7fb1 d407 724c 5035 a335 109e 791f a4b7 03f2 d8be 9a8e 031e bad6 7175 90c7 ad03 9250 9f4b 177b 40f6 0653 9eb7 6d1d 49e8 949e bc12 989f ad28 675e 8dd2 eb59 e5fc 4703 2ef1 b9a1 da84 b8d1.

J_{A6} = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e206 9341 2a6d fe70 276c 2465 ee00 e206 9341 276c 2465 f078 5c20 9341 2a6d fe70 276c 2465 ee00 e206 9141 276c 2465 f078 5e20 9341 2a 6d fe70 276c 2465 ee00 e266.

C_{A6} = 1ae7 264d 6b92 9c8d 3131 5411 e0b0 65c1 9ac2 c815 a6dc 92bd 26e8 2281 8ce8 d9b0 bde2 b895 a267 f6eb 226e 3898 fed6 fac0 1865 66f0 9bed 5992 b882 02c1 2df7 e903 3849 4881 8570 298d 8df1 27c4 4758 f769 ccf4 a6ce 2303 3fb0 b130 17c9 8eb7 7db4.

J_{A7} = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 ef07 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 ef07 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 ef07 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 ef76.

C_{A7} = 496f db6e e535 0180 1c86 6610 8769 f225 4631 6b07 0267 d57d 7613 f42c 70e1 0d39 256f 5c60 3d22 7f28 5401 0d42 05d3 eab1 f795 b386 5595 4234 7f8a 3ce0 b483 8d7f 0c7f 0cf2 dea6 f895 277a a2f5 1732 e854 8f44 d31a e502 ad03 7194 0d65 ab07 c55d c85f.

J_{A8} = 29a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7004 49a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7004 49a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7004 48a0 93b6 1232 f83c 2f10 2f10 49a0 9536 ff38 13b6 1232 f700 7043.

C_{A8} = 4892 3caa be40 0643 ce8a 24ab 1e48 5876 ae94 9ae1 a465 ced2 a59e 63d2 fd37 044b 202e 1543 64db 38b5 d16a b675 2401 98be 23c1 ba7c ca8a 0058 4ae9 e637 f70a a640 c1a8 a1b3 2f5a 3a35 7e75 7df4 04e3 0ffb 8203 0b5c 986a 131b 1aa7 b37b 2c04 4dbe.

C.1.1.4 鉴别交换

鉴别程序重复 $t=3$ 次。

迭代 1：

步骤(1)：

r = f637 29e2 8723 3b12 d7c9 b048 8626 7680 3880 f8b8 0ba0 3497 60fe 2c2d ee4f a8ed 7860 8a3f 24f1 22f4 45d1 cb18 8ef2 82c1 a6ea 4453 c550 cfcf d16f ebdb add9 51ed 750e d717 cd83 8cd3 1cbc ce82 c2e6 afe4 8507 e66b 5417 33eb d4e1 8c94 e180 e1c6.

W = 573c 7004 78cd feac 4025 20fa 7e68 8010 deb9 0a10 676e e59b 9074 40be e961 86af f988 6449 1a34 aa7e b741 1af9 3e12 fd4b d9ff fc0b dfc9 5c1a 3780 8b77 aa8f 3170 e240 c6ae 9d03 c7d9 9acc e21b f125 a4f7 9f16 f26d e6bf dcb2 d6ae 6187 d536.

步骤(3)：

$d_1, d_2, \ldots\ldots, d_8$ = 0,0,1,0,1,1,0,1.

步骤(5)：

D = 415c d169 1334 412a be2d 6bdd d373 305f 0eed f1d2 f943 68e6 62cb da0e 9e46 8840 af85 9372 8fb5 6354 3bb5 7061 ff42 210d 6fbf 8232 3ef6 82a3 1956 2d57 3e06 0a6c 5c63 9762 a18a f0ca 12de e2bf f91d c376 844a 7f14 6113 c6a3 269c 8211 2d62 cbc5.

步骤(7c)：

W' = 573c 7004 78cd feac 4025 4777 20fa 7e68 8010 deb9 0a10 676e e59b 9074 40be e961 86af f988 6449 1a34 aa7e b741 1af9 3e12 fd4b d9ff fc0b dfc9 5c1a 3780 8b77 aa8f 3170 e240 c6ae 9d03 c7d9 9acc e21b f125 a4f7 9f16 f26d e6bf dcb2 d6ae 6187 d536.

迭代 2：

步骤(1)：

r = 6d5f 2b2f 5027 39a9 177d 30f4 8d5e 1b6d 40d4 eea9 35d8 3bb4 0288 b447 6cdf 3f5c f0f9 b714 08dc eaae ff5a b380 c96b acb0 973b 78ab 08f6 d085 795d 92e3 630b ffa8 59ae 1eb5 2b52 0c5d 6030 fe80 a4e1 1a2f ed3d 6801 5311 a0ab 5018 8a73 f1b6 9460.

W = 44ce 41f4 9c42 ec82 f5ac 5a42 03e6 6cc5 bcb6 f343 f0db 0c07 5318 fc26 328f 1f07 35df 653b c8bc 9bf9 a6af fafe 98a5 14d6 4952 1f5b c1a9 de55 721e 8950 399d 7c3b cb2d ba2d ef50 5cce a17e e8ec 314a 6621 ccd4 5755 5136 b4f6 fdb6 ce3d 94c9.

步骤(3)：

$d_1, d_2, \cdots\cdots, d_8$ = 0,1,1,0,0,0,1,0.

步骤(5)：

D = 0c84 7739 615a 2c11 a240 6314 bb0c 1bce 155f b99e 7c92 8c3d 15a8 ee81 ea67 1f6d c2248 d1c8 9c06 a80f b4e2 a2cb d5e3 54ca b5ba 98f6 f2ac e1f6 c160 dc82 8f8e db4f 2131 a11c 86a0 86ba 9f74 e838 b653 cd25 02cc 1877 1ed7 113c 1cb3 bf19 6cac 738f.

步骤(7c)：

W' = 44ce 41f4 9c42 ec82 f5ac 5a42 03e6 6cc5 bcb6 f343 f0db 0c07 5318 fc26 328f 1f07 35df 653b c8bc 9bf9 a6af fafe 98a5 14d5 4952 1f5b c1a9 de55 721e 8950 399d 7c3b cb2d ba2d ef50 5cce a17e e8ec 314a 6621 ccd4 5755 5136 b4f6 b6de ce3d 94c9.

迭代 3：

步骤(1)：

r = c50a 4b30 b2ad 7b7a 26ac 6ca5 0b2e 2d2c 0d40 1fb8 4e6d 6d12 3fce c8f2 9f55 26cf eced cbf0 184c f826 5db5 db87 a82e 9397 9fd5 9152 b65a fdbd f5a2 c017 9781 33ab 12ec f85f 5db8 9fdb 6aa5 43b5 87b2 88f0 2963 4604 9703 5838 7cb3 28bd a1a9 b699.

W = 0268 2fb9 c79b 2c9f bdc3 2804 6dc5 9a30 d3c7 0e02 01db e43e 2c09 8fde f967 037f 20be 354e c92d 208e ecfd a688 7126 58ef 28fd e27c c97b 8520 a408 1570 0539 de84 632b 0ba2 b899 95c9 199e 9d61 a0cb c036 2ed1 0a8e d566 4935 98cb 7f32 038d 525d.

步骤(3)：

$d_1, d_2, \cdots\cdots, d_8$ = 1,1,0,0,0,0,0,1.

步骤(5)：

D = 3c61 982e fa0e 5c75 dada 504e d5e2 b056 e3cf 80bb fad1 2925 05bd 426c 952e bace ffd3 cdb3 cb5d 2233 7128 9507 81a0 464a f7cf db3e dbaa f76f 2dla d3c8 7ce2 5289 fe4d 87eb 9683 20ba 749a 369a da50 0227 bd8a 8439 8e8c 5a4e 82eb 90a1 0294 2448.

步骤(7c)：

W' = 0268 2fb9 c79b 2c9f bdc3 2840 6dc5 9a30 d3c7 0e02 01db e43e 2c09 8fde f967 037f 20be 354e c92d 208e ecfd a688 7126 58ef 28fd e27c c97b 8520 a408 1570 0539 de84 632b 0ba2 b899 95c9 199e 9d61 a0cb c036 2ed1 0a8e d566 4935 98cb 7f32 038d 525d.

C.1.2 公开指数为3的例子

C.1.2.1 参数选择

在这一例子中公开验证指数 v 为3，一个奇数。因此秘密的素因数 p 和 q 必须满足：

$$\gcd(p-1, v) = \gcd(q-1, v) = 1$$

p = a0ca e977 6bc5 5a7f f591 7bc8 8164 16f9 503c 16e9 0a0c 4da3 b1d3 d97a 1220 605e 071f 1c6f 9305 def5 4832 0ea3 5e76 4d45 698e 9196 09a4 35f1 fde4 0d7c 3146 8eb3.

q = e349 3f5b 7808 aac9 6083 b0b6 d97d 5a57 d300 43c8 6416 719e 2d95 7654 5f0a c7b1 4061 8232 728c 7777 0fbe aac2 f5f0 8238 5783 91bb ceb5 be1e cd31 b043 be4f 75df.

公开模数 n 为1024比特长。

n = 8ec1 eeac da97 aa8b a6e2 fb76 4423 dcc7 d723 2848 5219 c685 bcef f9a8 f970 a9b2 ed4d 7dd8 64dd 162d f77a a9b8 549e 7029 d409 9494 61af 3590 e5ca 6cf4 1f5b 073d b399 f566 0068 4ff9 60f8 8336 85a6 d337 84c3 cade c2e9 32fe a1fe 9b05 85b2 8a8a 4b02 4bdb 7d46 9b62 2657 b19a ade2 768d 3608 f0be 09bb 9d59 c88c 7d3c 0eeb 1ced.

k_s = 1023

认可机构的私有认可指数 u 为最小的正整数满足：$uv+1$ 是 $\mathrm{lcm}(p-1, q-1)$ 的倍数。

u = 2f95 fa39 9e32 8e2e 8cf6 53d2 16b6 9ed 47b6 62c2 c608 c608 9781 e9a5 5338 5325 8de6 4f19 d49d 76f4 5cb9 fd28 e33d 718a 2563 46ad dc31 75e5 11da f743 79a6 b51e 57be ba81 eedb b433 6e3a ae4b c792 6397 20a2 2082 7ab9 c6ec d13e eb87 1912 5c2d 20d3 abd5 e468 7d3c 16fc 9a22 52bc 1dd3 e25a 7c52 4479 65cb 386d a9d2 3fd4 0a71 b2c9.

$$m=5, t=5$$

C.1.2.2 身份选择

标识数据由 $m=5$ 部分的序列组成。这些身份数据使用串“AlexAmple”并以一个16比特的字段数字为后缀构成。

I_{A1} = 416C 6578 2041 6D70 6C65 0001

I_{A2} = 416C 6578 2041 6D70 6C65 0002

I_{A3} = 416C 6578 2041 6D70 6C65 0003

I_{A4} = 416C 6578 2041 6D70 6C65 0004

I_{A5} = 416C 6578 2041 6D70 6C65 0005

C.1.2.3 认可产生

J_{A1} = 766c 2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341

276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9141 276c 2465 f078 5e20 9341 2a6b fe70 276c 2465 ee00 e316.

C_{A1} = 2f98 686f a57f 0799 f0a6 42dc 20ae 91f4 f875 a346 a6b8 e951 042e 77c6 1ad9 0a60 915d 8ea6 9dbf bec2 589f 331f 26a7 0859 9ca8 27b5 5a5c b78b ee4e 07b4 9c86 dd7e afea 5a4e 9b9d 068b 173d a27e ebc0 78b1 8305 e930 48db b81e 49cd 83f0 e101 260a 76bc 10d8 8679 5b4c b096 7943 5195 ea43 b98b 35a4 c3b1 eb02 8e7c 0f54 0949 67f8.

J_{A2} = 676c 2465 ee00 e502 9341 176c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e502 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e502 9341 276c 2465 f078 5e20 9341 2765 f078 5e20 9341 2a6b fe70 276c 2465 ee00 e502 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e526.

C_{A2} = 31ca f31b 374b a31c ba12 b38a 23e8 46c7 dac1 7ead 4647 34e9 8adb 781b 4c56 243 ced6 9ee3 6eae 0992 c2af 5027 5874 7b15 4f2d 723d a590 593a ec39 5fbd de29 7bc5 0fab 1af9 7143 a0d1 b25a b1bc 6c7f 544f 72a9 35ec 265d a54d d52d c5b6 cde9 58ef 583b 2dfa 5ba6 0c05 8f11 6c9f 488e d9fb 8680 112c ba35 aec8 7441 7aa5 8f4b 57f5.

J_{A3} = 676c 2465 ee00 e803 9341 276c 2465 f078 5e20 9341 2a6b fe70 276c 2465 ee00 e803 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e803 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e803 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e803 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e836.

C_{A3} = 2dac 4f99 c5c7 6a01 56bf 01f1 d135 2b07 3742 cc23 9e43 eaed d6d3 0e9f 8c17 750d 0024 5099 8caa 7c76 526c adc6 cd78 74d9 90b3 bcbb abc0 603f 0431 ae82 81f0 2d92 4c8b 3add 7eb0 2c37 bca5 2ea5 c740 6752 0bdc 0644 a64e ae8e 6690 4ac0 31af cc8d ac8b 1a4f 8c04 de6a bb29 3d98 7449 0a87 56e6 c54d 0259 07a3 136c a560 3c42 d0a1.

J_{A4} = 676c 2465 ee00 e904 9341 276c f078 5e20 9341 2a6d fe70 276c 2465 ee00 e904 9341 276c 2465 f078 5e02 9341 2a 6d fe70 276c 2465 ee00 e904 9341 2760 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e904 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e904 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e946.

C_{A4} = 10d9 1f22 933c 27c2 589b b1d0 0a27 767c 1465 b9de 1074 aaf5 e845 128c d422 8cb1 2688 7139 2526 7d7d 6e07 cf01 975e aeac 9b50 9e8b 2432 06de b57c 0676 5069 5ddb 2a53 419d 7703 30ca f37a e3d7 148e 8dab b8c9 dd68 359a cb64 7d4c bbf1 156b 5aa8 49f0 7e12 3e0f d71c 853a d005 a36b 9bf9 ee65 3e12 cc18 4fa1 a3b8 9ee2 a121 353c.

J_{A5} = 676c 2465 ee00 e405 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e405 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e405 9341 276c 2465 f078 5e20 9341 2765 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e405 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e456.

C_{A5} = 171d fcf1 dcda 63b1 a53c 1f8b a8a7 d46c 8ced c942 956d 9fea 5469 61dc cc52 dcd8 4821 604e bde1 cd25 d23d 7815 0bdc 85b3 7aa0 b691 9609 6aae 6ca3 e843 5575 4eb1 2470 234d 05ea 8a46 45c2 fa69 eaac e374 78d2 7394 12db 57c7 9018 6a85 8a00 8e56 732d 71f3 f5ea dc8e 0988 95cf 3a50 cfd8 e106 b604 2c9c 9c07 c630 6e15 ead4 e050.

C.1.2.4 鉴别交换

迭代 1：

步骤(1)：

r = 0fdf 125c 140e 2a4f 54d4 ef3b 41a4 e11e 7175 4f7b f003 1d01 8a51 8ac2 27be 49fa 6987 e987

ebb1 0ebb dbfd ee1a 9443 7313 fc53 44e0 5238 0a7f 155a 9ad0 70fd e4ef db37 e6fb 1a82 d078 bc73 31a1 b95b 58e2 3665 ce63 1300 e862 2d52 5552 dd45 fdde e10e f0e5 5d63 106c 0692 5dd8 4b1c ba98 fa0c 4e57 1a0b 135e c5d2 9659 47bf 7145.

W = 2935 3492 54b4 b32c fa87 b7af 0a17 12ef 568b c5b9 593d f8a6 2972 b911 18c1 9d2e 5cee 57a5 1318 d878 e829 e9e7 0306 f482 e81f 57e4 e18d d71d 5312 a0a7 3539 971a e02f 46b4 1e8a 456e 9260 4090 8018 9c7a dfea d147 d75c 2aea 143e d666 c3b4 3993 616b f040 c310 7baf 4584 c2e7 ec50 b8b2 040d c803 19c4 4faf 7b00 b3b0 67ab 8ee2.

步骤(3)：

$d_1, d_2, \cdots\cdots, d_5$ = 2,1,2,1,2。

步骤(5)：

D = 2381 3185 d61b 1684 1c72 8b6d eb9e 30a6 6b7f c509 3c00 c5de 6194 9ba2 9721 a8df fe0d 04e5 f5c5 2e71 ac97 b511 a144 d4e9 7d5b 359d 1e92 ffb9 ca3a 2e61 42a8 a0a2 b3cd 22ee 1e28 5432 d645 21d4 73f6 d9b0 5765 22fb 3ea0 78ba 1ee4 f70c 8d56 d354 df85 a7e3 1257 f9c2 2865 d0e9 2f8a e587 61b0 4f04 4dd2 8b74 cf60 b332 d461 e5c7.

步骤(7c)：

W' = 2935 3492 54b4 b32c fa87 b7af 0a17 12ef 568b c5d9 593d f8a6 2972 b911 18c1 9d2e 5cee 57a5 1318 d878 e829 e9e7 0306 f482 e81f 57e4 e18d d71d 5312 a0a7 3539 971a e02f 46b4 1e8a 456e 9260 4090 8018 9c7a dfea d147 d75c 2aea 143e d666 c3b4 3993 616b f040 c310 7baf 4584 c2e7 ec50 b8b2 040d c803 19c4 4faf 7b00 b3b0 67ab 8ee2.

迭代 2：

步骤(1)：

r = 583e 235e d777 a918 1d4a a2f9 da89 a905 3e65 a827 d573 c68e 73e7 ce5a 61b1 d135 1d6c 4236 df99 2c1a d174 588b 8e07 5f14 c954 44cf 3493 b44e 3971 dadc 38e6 3a86 a1c8 e4dc 222e 5290 86a3 3301 cd87 5be2 3c5b d963 df15 3f4b add7 4ea5 3eb5 f7a5 41d3 90a0 99b3 78ad 46ac 635a 3bdf 72eb 959d 4e1c ca8c 12aa 4a34 38ce a016 556f.

W = 2f69 40df a78e 740f bf64 9f12 f55d 5081 ab45 d794 b08b c3f9 98e d1de f793 7f93 7fa5 9b20 860b 9474 bd0e 4874 93b4 ec52 6c52 6c31 7e2f c250 3a4a 4951 1b0a 9b12 8e53 80a8 b58d beca 35c2 d633 5b80 f184 89fb eed9 b3da f95f 71b2 0eb1 f6e9 c4f6 2054 7eff 0ac5 97b5 7fc3 7d89 e746 91d1 8417 440a 1d37 63fd 377d 09c7 5369 5369 88c5 9389 d3f3.

步骤(3)：

$d_1, d_2, \ldots\ldots, d_5$ = 1,1,0,0,0。

步骤(5)：

D = 10bf ace9 3bc3 80e1 f8a8 9229 f66f d7af 5819 d745 be0b 7980 8df0 0692 2e8f bdef 8650 d311 2269 eae1 7bc2 a641 509d 1238 148d d07c ca97 3d1a 67c5 e34c 50e2 9ab8 52eb 42e7 e527 6eb3 6cfb 18dc ec2f 6939 9461 7f05 6320 94b7 8c07 87cc db53 8f85 d8be 6754 e39b 9f2e 30cc 935f c2da 97cd 60c6 99ba bfac 07fe 02ea 406a 613e 34ea.

步骤(7c)：

W' = 2f69 40df a78e 740f bf64 9f12 f55d 5081 ab45 d794 b08b c3f9 c98e d1de f793 7fa5 9b20 860b 9474 bd0e 4874 93b4 ec52 6c31 7e2f c250 3a4a 4951 1b0a 9b12 8e53 80a8 b58d beca 35c2 d633 5b80 f184 89fb eed9 b3da f95f 71b2 0eb1 f6e9 c4f6 2054 7eff 0ac5 97b5 7fc3 7d89 e746 91d1 8517 440a 1d37 63fd 377d 09c7 5369 88c5 9389 d3f3.

迭代 3：

步骤(1)：

r = 170d 20a2 00f4 7124 fac0 695d 1612 07ce 848a cba5 d1f5 adcb 0ab1 ae12 6377 4b5a 4ea1 48e2 434a 804b e374 6787 a775 29c9 59de c8c5 2952 cade 4d81 3061 6f6d f1ba b7b6 2b24 be92 5cfc 44e6 e2df 2b97 31c2 180f 3b32 c068 ee9f 0b70 88e0 e700 67d9 d799 0622 b329 bea3 c1f3 ff80 e614 3562 794a a065 28e8 61dd fc41 109c 6080 ded0.

W = 2c81 4f76 98b9 4169 b7c5 14a4 6f3c d9e9 5f7c 7aa1 9de9 9d59 fd42 7d8a 6f96 a45c 345c 823e e7ba 6dee 98a0 2c44 34b9 4249 32df 3ef8 4f79 4f8a 69d5 c970 a5f8 f8bc f1f7 4189 56a9 e9aa ae01 7928 2ccf c1c8 5c75 1a7f a117 6c66 366d 8ee2 ef5d 9145 0aa1 18e4 1bc1 6601 373a 1340 72ba adb6 1565 e292 d826 0047 afb8 0b20 f227 2290.

步骤(3)：

$d_1, d_2, \cdots\cdots, d_5$ = 0,0,1,0,2。

步骤(5)：

D = 10fc 250c 11ad 61b8 19b1 eaa3 9376 4e22 cfce 7b98 af17 0d1e b74b a5c7 596b dafc f953 7fa6 c6d7 6fe9 61c1 c7bf 087b eb8d 74d9 ce81 959e f419 1912 927f da1c 04f7 92a6 dd1a 0ca3 7f86 2dc6 c4b9 331f 69a7 cd46 c7ad 7352 43d4 6223 ba0a c578 aaf4 1443 a413 67e7 5497 47e3 7603 ecad e131 f18a 771c edd2 064c 5d56 11a9 7fa1.

步骤(7c)：

W' = 2c81 4f76 98b9 4169 b7c5 14a4 6f3c d9e9 5f7c 7aa1 9de9 9d59 fd42 7d8a 6f96 a45c 345c 823e e7ba 6dee 98a0 2c44 34b9 4249 32df 3ef8 4f79 4f8a 69d5 c970 a5f8 f8bc f1f7 4189 56a9 e9aa ae01 7928 2ccf c1c8 5c75 1a7f a117 6c66 366d 8ee2 ef5d 9145 0aa1 18e4 1bc1 6601 373a 1340 72ba adb6 1565 e292 d826 0047 afb8 0b20 f227 2290.

迭代 4：

步骤(1)：

r = 4224 ffa1 41bc 4bea 93d0 14ec acc1 fa5d 9616 f0fa c12a a029 3f84 a858 f916 13c3 ccef e0d1 16e5 30fc 4fda 72f5 15ce 5996 e211 fcc8 eee0 0719 84a6 b717 a7be cc05 afd1 8b8e 71ec 2d3f 285c 6d07 39c6 b4ef 3660 468d c13f 24c8 0ef8 c992 59f0 04c8 996e 9387 99a7 0769 03d7 fd23 9472 3396 6cc0 2ec3 fc30 33b9 4a8c d1fb 919f 610a.

W = 0525 c5ba f6f9 78b1 1fd6 6e86 0de3 ebd3 314e efb0 9ca7 8193 3b2e cb5f 8046 1b46 a87a 727a a317 e163 9dc4 2b55 8202 65dd 0f2c ce4d 57fe 84dc 08f5 49ee 896a a897 c29a dab e0c6 78fe 0be5 753d 0d97 0d99 f3d2 1eef b822 7715 39e7 402d 63da 03d4 b66f 3789 066e 4c36 d025 3466 1b6a e359 f290 bb21 c1fc 01fe 23fb 0135 77d8 0ea9.

步骤(3)：

$d_1, d_2, \cdots\cdots, d_5$ = 0,0,0,1,2。

步骤(5)

D = 3559 cda5 4d03 1913 3dca c484 3a3f 9635 86e1 8455 3448 e759 1213 5269 9d8a 5a5d 4c4e 178e befd 7223 2808 356a 427a 0b1e 1681 3ba4 c564 27bf 0c03 1fb9 e35e ebcc 51c3 bfb2 346c 11f6 bc7c 3dfc d3f5 000d 2614 1ad5 7197 4362 f195 9750 a8ae 5750 f36d fd40 b930 85ca 4600 b753 0e36 9791 4c87 4860 56a9 ad71 4717 eebc 3dc3 633e.

步骤(7c)：

W' = 0525 c5ba f6f9 78b1 1fd6 6e86 0de3 ebd3 314e efb0 9ca7 8193 3b2e cb5f 8046 1b46 a87a 727a a317 e163 9dc4 2b55 8202 65dd 0f2c ce4d 57fe 84dc 08f5 49ee 896a a897 c29a dab e0c6 78fe 0be5 753d 0d97 0d99 f3d2 1eef b822 7715 39e7 402d 63da 03d4 b66f 3789 066e 4c36 d025 3466 1b6a e359 f290 bb21 c1fc 01fe 23fb 0135 77d8 0ea9.

迭代 5：

步骤(1):

r = 0d2c cc3f 814b a506 7753 4948 8ba7 c8de 8a06 8da8 2eba 8b9f fc7e ab4f f439 8317 91b0 2700 28c8 c170 e8ec 16ca f279 08ba de15 912f dbf4 0562 0b1d f95b b16e b24a 2b46 e0d7 e466 c636 684b e07d d57f c67b e66d 6b16 af88 d873 047a 94e1 be1c 639f 2426 efb5 614b fbeb 00c4 d6af 0d14 0ac5 d54f c632 f933 80e6 e94d 5a35 675f fa6d.

W = 23d8 0778 c35c eccb b47f 8d83 881d 5e18 dfb8 a4e7 eeaf e9c8 af80 ef7b e6c0 a9b5 3d3c a4f3 524b f72c 18b6 edb7 ba71 4523 9d48 4463 a817 9f18 83cf 71e9 96dc da61 58a2 3936 91bf 133b fdfe e906 b713 64b6 cdef 8446 2be2 8634 67d0 0f78 7a67 9ba4 b42f 92d1 ea7d 8c5e bfb9 8b7f d049 3907 61a8 1300 a9d1 4e18 cf4b dc08 064a ddcb.

步骤(3):

$d_1, d_2, \cdots\cdots, d_5$ = 0,1,2,1,0。

步骤(5):

D = 1061 f560 095a f8d8 81ae 7844 aa56 62cd d651 4c40 4f1a 4cb1 ebc3 0672 7ccc 4c10 0ded 8447 e1b0 5611 7135 78f3 86fd 6251 59ef 5755 c26f 7780 9fa9 a601 d059 b755 4ccf9 bfa8 f491 6ce4 96c1 c082 c9d9 8622 b035 4341 3171 fa8c 6be3 c07a 418c b828 02f3 05a2 6998 40dc a632 ef9c a5a8 c9b3 e81e 902d 6699 4c00 4efa 3ce2 766d 42c1.

步骤(7c):

W' = 23d8 0778 c35c eccb b47f 8d83 881d 5e18 dfb8 a4e7 33af e9c8 af80 ef7b e6c0 a9b5 3d3c a4f3 524b f72c 18b6 edb7 ba71 4523 9d48 4463 a817 9f18 83cf 71e9 96dc da61 58a2 3936 91bf 133b fdfe e906 b713 64b6 cdef 8446 2be2 8634 67d0 0f78 7a67 9ba4 b42f 92d1 ea7d 8c5e bfb9 9b7f d049 3907 61a8 1300 a9d1 4e18 cf4b dc08 064a ddcb.

C.1.3 公开指数为 $2^{16}+1$ 的例子

C.1.3.1 参数选择

在这个例子中公开验证指数 v 为 $2^{16}+1=65\ 537$,一个奇数。因此秘密的素因数 p 和 q 必须满足:

$$\gcd(p-1, v) = \gcd(q-1, v) = 1$$

p = b843 ab40 c311 80cd 1063 f5d6 2158 bc6d 9c93 b2fc 1def 7dfd 3152 a695 89d9 4a80 1000 bfee fb62 7312 5552 2138 61d2 39a1.

q = a8a1 c635 9063 d197 15a0 8e5f cd38 d6f2 3530 dde0 7359 2a67 1d02 e72a bb8e 8be2 599e 5bc8 ab53 c780 7d5e 9fd8 f680 5f79.

公开模数 n 为 767 比特长。

n = 7960 d99c 1822 9f2c 2607 75d2 2eae 9941 6942 b3e8 f5ad c612 cd3f c529 70ed a698 0ba2 6388 08bd b5cb c048 d63a 82d4 ac95 b166 9c43 4135 d7fc 19e2 022e a465 6bcc ee9b 7dba d90e 1125 91a2 1a66 1494 f4f6 e2b5 ce43 3b6a f390 79a0 2b48 c63f fc19.

k_s = 766

认可机构的私有认可指数 u 为最小的正整数满足:$uv+1$ 是 lcm$(p-1, q-1)$的倍数。

u = 02f8 d0c5 e0fe bd5a fd60 b80d 24c0 cdab d657 4b19 c1cf 8c1c 05be 86a5 ff1d 3289 0d2f e009 57fd 727d 6cab 3138 f989 c4e2 aa1d f4dd a901 f518 c560 7fa7 1ea7 4893 535e 1d2a a178 949a 0c25 cc59 25a6 25d6 337e 24c7 af7a 6fff 6d42 f61c d3ff 8dff.

m = 1, t = 1

C.1.3.2 身份选择

标识数据由 $m=1$ 部分的序列组成。这些身份数据由串"AlexAmple"并以一个 16 比特字段数字为后缀构成。

$$I_{A1} = 416c\ 6578\ 2041\ 6d70\ 6c65\ 0001$$

C.1.3.3 认可产生

J_{A1} = 3341 276c f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341 276c f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e316.

C_{A1} = 2c61 c981 f375 ed78 5a4e 9939 e054 c63a 3809 8f2f f525 ed20 2d4e 0a65 f7af 7548 80ce 954f 8f15 a1e0 bb73 9cbb 815c 5970 4f1c 4e3e 7552 dd1c 4966 d352 1992 149d f30c be32 d1c7 569b 40e4 8b7d b558 b003 95b8 2ec1 c1e6 3ed3 cfd9 abe0 22de 827c.

C.1.3.4 鉴别交换

迭代 1：

步骤(1)：

r = 3a2c 36ef 335d b967 a76d b60f 7ad0 a6ea 518a face 23d1 5ef3 3ead 46e 89af aa30 7a57 b5eb e1f5 b4aa 952e 125b 18be ac2f 245d f716 45e5 baee 9c5c 2750 4a76 92ac 0a45 bdd6 89a1 322f 6fa6 46d9 e18d deb9 c94 e791 50cf 0776 104b 6f1d e369 977a.

W = 22b6 a130 b77d 5ace b6e6 d55f 3eab d710 cd08 184d df53 6cd1 0372 b0da 7f34 97cf 9355 87c1 c877 f58b ddd6 5595 ec51 1e1f 7da2 5677 a22a 9e3d 6f45 23cf 19cf 1927 ac1e 76ff 6614 0d46 9f2d 44de.

步骤(3)：

d_1 = 003d

步骤(5)：

D = 1749 15ce a8cf cabb 678b 32d2 c8fa 636e cdca 7918 f47f b631 7741 e35e d84f 9257 bcd0 b8f6 27bc 7db4 b852 032b 8dea 028c 3681 a15c fedc ede8 8094 77c1 bec8 def4 c768 c78e 05f4 327e c58c c0f7 8229 8616 f15f 819b 819b 746d 0efb 4747 9581 b39d.

步骤(7c)：

W' = 22b6 a130 b77d 5ace 0eaf b6e6 d55f 3eab d710 cd08 184d df53 6cd1 0372 b0da 7f34 97cf 9355 87c1 c887 072d e166 c09f 6f3e 1b21 4952 6be0 6be0 774c e4fb a38f f58b ddd6 5595 ec51 1e1f 7da2 5677 a22a 9e3d 6f45 23cf 1927 ac1e 76ff 6614 0d46 9f2d 44de.

C.2 基于离散对数的机制

C.2.1 使用 768-bit 的 p，128-bit 的 q 和 RIPEMD-128 的例子

C.2.1.1 参数选择

此例使用一个 768 比特的素数 p，一个 128 比特的素数 q（$p-1$ 的一个素因子），并以 RIPEMD-128 作为散列函数。

768 比特的素数 p：

p = d716599e b22836ac fb221d0a f4c66b16 e3dceaee a73a17fb aaa33c07 6cf3571f 54d89d49 38d7c311 e24b98f1 e510599d b53f7387 0d2acf2f 8fbf8267 c1df4fe3 2e8a04e4 14125c5f d6d8efd7 8c5f1563 4288cd6a 0caaf4cd 3cf44434 d7ea8134.

128 比特的素数 q，($p-1$)[1]的一个素因子：

q= a73a17fb aaa33c07 6cf3571f 54d89d49.

Z_p 中阶为 q 的元素：

g = 5c7af3fa beff6338 f3137b85 a83e557b 49135e47 ba7ed438 e34b1fa0 af8c2651 15cf8b2f

1) p 被选择以便 q 的拷贝被嵌入到 p 内。在存储空间和/或通信带宽非常珍贵的情形下，这种选择 p 的方法是有用的。

3c924b33 0addf043 10ee6c41 a378541f 69a370dc b09f898a f3204864 8a8433be cdb55d5d 6cdbc85d b6f3a645 0df8b209 1a674d77 cdee3ele 86d4fb93.

C.2.1.2 **密钥选择**

实体 A 的 128 比特私有密钥：

$$z_A = \text{f1d4e85a eff74310 53adcac0 a9c155ce.}$$

实体 A 的 786 比特的公开验证密钥（$y_A = g^{z_A} \bmod p$）：

$y_A =$ 813eba80 bdbfdc78 15d09f74 ec21e1a6 4bd1a3c6 7a2591b2 44d53e2b 0c4dcb54 d5e2a8db 411e7c04 566d2671 12e72695 4a3c7699 a304255e fc58390c b0566240 66ee9d2b 131ebbca 7217f973 86dcab2b d4fb246c a2b5da9d 80954a65 4f6f3c6.

C.2.1.3 **鉴别交换**

步骤(1)：

实体 A 选择一个随机数 r。

$$r = \text{868b4b13 017364b7 e7dda29e cda55473.}$$

$W = g^r \bmod p$

= 6ae62a6 d172be24 85830f5c c1524f4c a23172a8 c8691011 759f63d7 86b1a7d7 a6809f43 512f42c6 a6a444d6 a437f62f 02881f99 6e3638b0 93f6da41 cd4860e2 fb856e58 8af85ad852 c5f90648 915498bc d47fe84a 621c7bf3.

步骤(2)：

$$h(W) = \text{f084606b 90de79de7902 2bb16d2f 31996976.}$$

实体 A 发送 $h(W)$ 给实体 B（Token$AB_1 = h(W)$）。

步骤(3)：

实体 B 随机选择整数 d。（此例使用一个 16 比特的 d。）

d=d47c

步骤(4)：

实体 B 发送询问 d 给实体 A。

步骤(5)：

实体 A 计算响应 $D = r - dz_A \bmod q$

$$D = \text{1edee4bc 1667f13a 7cbf9d28 c5a356ea}$$

步骤(6)：

实体 A 发送 Token$AB_2 = D$ 给实体 B。

步骤(7)：

实体 B 检查 $0 < D < q$，并计算 $W' = (y_A)^d g^D \bmod p$。

$(y_A)^d \bmod p$ = 6e6f703d ace31e6f 335f556b 42b24a6d 21771d60 2fa448be b74ae11d 3d63ff2f cecf2452 1417d470 04839f4a b15e91fa 72bbebfb 9b9c4b41 8c0c6a2d ef4a40e1 9a4a629c 32e63a0e fb03ec80 d55efeb8 173c6b26 58d28216 3be0ca6a 2d90cae6.

$g^D \bmod p$ = 8e249ba6 f0e2f8a 156c0851 0bf23a2f 3408b9c0 7ec733cc dcfe78cf ca3bf160 7217be06 ccd9abc5 a392c7e7 482307c 2e7c310a b26523a7 af58361c a685c3bb 1cd38f91 15479db1 cf38f7c8 852a4644 14cdc936 797e6883 7106bcb6 d1bbeaf1.

W' = 6ae62a6 d172be24 85830f5c c1524f4c a23172a8 c8691011 759f63d7 86b1a7d7 a6809f43 512f2c6 a6a444d6 a437f62f 02881f99 6e3638b0 93f6da41 cd4860e2 fb856e58 1a2cafed 8af85e6c 856ad852 c5f90648 915498bc d47fe84a 621c7bf3.

如果 $h(W') = h(W)$，那么对实体 A 的鉴别完成。

C.2.2 使用 1024-bit 的 p，160-bit 的 q 和 SHA-1 的例子

C.2.2.1 参数选择

此例使用一个 1024 比特的素数 p，一个 160 比特的素数 q（$p-1$ 的一个素因子）和 GB/T 18238.3 的第 9 章中所说明的专用散列函数 3（也即 SHA-1）作为散列函数。

1024 比特的素数 p：

p = ea9b8f92 26d7b2f6 729122ef 53ce81e2 687acf40 a7db660e ba5e4daf cb0ebc3a ccb15c36 896f67f0 703e7c69 afc4c24b 221a8968 5cdcfb3e 086d8f95 702cbfc5 8e4170a2 e10df7b5 2bf8f015 c5a689ca 48df291b e796c443 f5e7ad19 8c159f0a ba9d962e 60d34840 77b5993e 48bbc3ed fef5f54c accde46e 69a3f1f6 1ae08af9.

160 比特的素数 q，$(p-1)$[1]的一个素因子：

q= cb0ebc3a ccb15c36 896f67f0 703e7c69 afc4c24b

Z_p 中阶为 q 的元素：

g = 26324f69 934e 6733 c66367a5 af5a08d8 455a5125 29882857 b20083e8 f72420a9 1f16a377 6dc612ff e652a2dd 05d51441 5f52c591 e8aa3127 8309ce2b ca9e5b73 5e8cc526 0dc1608d 91f32a8d 31265adc f2f2ff5f a4a786ef 25086bdb 061355cd 96ea33f6 429aef56 bc0c0aba db1ec3e0 b1140687 d60678c6 205c7f6d 6a236f87.

C.2.2.2 密钥选择

实体 A 的 160 比特的私有密钥

$$x_A = 87146299\ 068b4b13\ 017364b7\ e7dda29e\ cda5547e.$$

实体 A 的 1024 比特的公开验证密钥（$y_A = g^{x_A} \bmod p$）：

y_A = 819b36e6 62ddc4af 146dcf3a f888d61b 560ea5ea 8bb368f7 0e822e95 ef5e45c6 68b98732 725d29dc 21bf1394 29d95de2 98a6d595 9a7188c3 ab4b5d6d 20ca1d9e d6bc4d7a d23a4e3b 48cbe4ac da28d927 922c85ff db7e1f59 71a17dd5 dc68725c 32cf50f0 be5d8a73 f93bf113 1c55bf51 35b314be 5067fd31 9867041d 4c96e5cf.

C.2.2.3 鉴别交换

步骤(1)：

实体 A 选择一个随机数 r

$$r = 87146299\ 068b4b13\ e7dda29e\ cda5547a.$$

$W = g^r \bmod p$

= 397ad6f9 b435b01b 4c43a2d1 008ddade 1a086c2f 0ea25134 ff5a8653 a374dfbf 47f1a543 fbb58232 0357ccel 33aeb861 6aebd4b7 65dea271 0dff3a09 7c40602b 7e719499 0e9c7717 0ce73286 930e9e27 f8053b28 d2c80fd2 ec529839 27f34f46 bb9842b0 bd9c6405 1b2c58d8 c5cdcc50 69c4a430 d0f93cd0 6f2f75f3 298684f6.

步骤(2)：

$$h(W) = d3cf43cd\ 80f2525d\ 360bf266\ d11590de$$

实体 A 发送 $h(W)$ 给实体 B（$\text{Token}AB_1 = h(W)$）。

步骤(3)：

实体 B 随机选择整数 d。（此例使用一个 40 比特的 d。）

d=a2 cda554a6

步骤(4)：

1) p 被选择以便 q 的拷贝被嵌入到 p 内。在存储空间和/或通信带宽非常珍贵的情形下，这种选择 p 的方法是有用的。

实体 B 发送询问 d 给实体 A。

步骤(5):

实体 A 计算响应 $D=r-dz_A \bmod q$

D=354bf25c 5f0e8cca f2aea2b9 7716a2d5 cb8ceb7e

步骤(6):

实体 A 发送 Token$AB_2=D$ 给实体 B。

步骤(7):

实体 B 检查 $0<D<q$，并计算 $W'=(y_A)^d g^D \bmod p$。

$(y_A)^d \bmod p$ = d95931d9 4ecd8e38 0993cf3d 9ab03767 abc0a08b 69a82166 83f73785 b940610f 9293ee53 be9e717f 6fd6a9be f7b0c140 1f374427 86856c96 c168f499 86800ecc 91f12765 be056ecb 7d03ce6b 4334a4b1 29cd1829 6705f4a6 105752c9 31190fe4 1a65c010 be4537f7 6913d471 50441aab 387a7e55 86e1debd 6343703f fd0eeef7.

$g^D \bmod p$ = c40924be 47db63b6 c48734a5 dd2f8a01 dc6c08ed 6cbfeda2 81b64230 fbbde7f8 fbdd-bd3e e64d6887 014b5b0a 78c0d111 c6550c01 01f00536 304bc91d 7efe0c1e f9dede7b 004534e0 74347241 b430ba21 bd1c2f93 903860b7 d1a14716 cc541c51 ade947ef 827e6a27 78d67db6 2b4db4ba 918ef0f8 7ccca628 25988779.

W' = 397ad6f9 b435b01b 4c43a2d1 008ddade 1a086c2f 0ea25134 ff5a8653 a374dfbf 47f1a543 fbb58232 0357cce1 33aeb861 6aebd4b7 65dea271 0dff3a09 7c40602b 7e719499 0e9c7717 0ce73286 930e9e27 f8053b28 d2c80fd2 ec529839 27f34f46 bb9842b0 bd9c6405 1b2c58d8 c5cdcc50 69c4a430 d0f93cd0 6f2f75f3 29864f6.

如果 $h(W')=h(W)$，那么对实体 A 的鉴别完成。

C.3 基于可信公开变换的机制

C.3.1 使用 767-bit 的 RSA 和 RIPEMD-160 的例子

C.3.1.1 参数选择

此例使用 RSA 作为非对称加密系统的算法，GB/T 18238.3 的第 7 章中所阐述的专用散列函数 1(也即 RIPEMD-160)作为散列函数。

我们假设 A 使用 767 比特的 RSA 模数 $n=pq$，一个公开的 RSA 指数 e 和一个私有的 RSA 指数 s，其中：

p = cef2 8973 ddff 2ad1 ba38 4a98 71e0 7de1 d8ad 973f e2e1 2d6d 357c 19b2 7304 79b6 5c7e 6369 9a25 bb49 9f41 e7de 0f6f a105.

q = 9327 da68 0aa9 a22f 201b 429a acf1 de30 382f cb01 cf3d 6b4b 85a1 fa3c f851 4738 5100 09ee 7dad 2b4c 4673 0971 a417 41ad.

n = 76f5 7c6f 1d53 742a 45a2 adad b9ca 6f4c eb1c 2317 6b02 9967 9ba4 3305 2c42 3146 44a3 9a79 5b57 5979 0685 8a10 932c f80d 8973 ecb6 30bf bc18 29db ff50 adf2 4465 a87c d236 1e8a 16c5 34f7 7acf ce94 0f59 234d c833 d279 0ade e26e 395f 71c5 1561.

e = 4d97 cc58 6372 3582 ae31 9d48 5814 05fr 4e74 1737 6710 f052 acda 8fd5 1827 b485 d762 c713 4bda b4bd 08d6 7f78 4a5c 174e d35b 5ff9 62eb 1965 6af3 a353 4635 e843 89a0 78f0 e042 e656 ff35 0236 33f4 cc86 fcbd 4349 7c0d 3c19 7d20 fc60 bc91 1017.

s = 678a a11c 459c bd6d 10b9 9555 675a 7d7c 7850 af9b dc56 fe01 8846 7529 d02e 4aa6 85d0 3d2e 0e8e c027 ba69 3b4c f4dc 48c0 f2ad 74a2 25c4 fc38 ec52 94e4 a222 d922 7d53 389c c95f 9410 2b00 5840 247b ea8c 7a1f 3c60 ac3a 7297 704c 36e2 afbb e107.

A 的公开加密变换为 $P_A(r)=r^e \pmod n$

C.3.1.2 鉴别交换

步骤(1):

实体 B 选择一个随机值 r。

r = adec c15b d356 b0cd 1b74 9469 b421 ac9d 28ee 96a5 85ec 6284 9cc2 8beb 0e59 4c9f 7377 f151 18fc a3df 249a b0aa ebb0 cf35 91f6 7858 d8a0 16e4 40bf ee20 bbcf da92 8e09 6e2a a6ec eccd f7f1.

B 计算 $h(r)$, $r \| h(r)$和询问 $d = P_A(r \| h(r))$。

$$h(r) = 4bc6\ 0e2c\ bfcb\ 238a\ 4d88\ 8b5f\ 3d4a\ eb02\ 3c16\ 1893.$$

$r \| h(r)$ = adec c15b d356 b0cd 1b74 9469 b421 ac9d 28ee 96a5 85ec 6284 9cc2 8beb 0e59 4c9f 7377 f151 18fc a3df 249a b0aa ebb0 cf35 91f6 7858 d8a0 16er 40bf ee20 bbcf da92 8e09 be2a a6ec eccd f7f1 4bc6 0e2c bfcb 238a 4d88 8b5f 3d4a eb02 3c16 1893.

$d = p_A(r \| h(r))$ = 60bd 94a5 7b0e 23b9 2eb9 0afb 5e21 630b 763f 8771 3479 98ed 4df6 3c9f bbf1 2369 d117 1c81 9277 63b9 3d5f 2af2 6288 7ee8 b24f b9d4 db2e c206 99b6 eb8f 2b31 3944 27e0 1f74 d501 cfca d45f 25ab dfd1 b605 c640 7d1a 597e 204a 1183 4670 c6b8 c52c 7cc3.

步骤(2):

B 发送 d 给 A。

步骤(3):

A 执行下列计算步骤

a) A 通过以下计算恢复 r 和 $h(r)$:

$$r \| h(r) = S_A(d) = d^e \pmod{n}。$$

b) A 根据上一步恢复的 r 值,重新计算 $h(r)$,并与在上一步恢复的 $h(r)$值相比较。

假定两个 $h(r)$的值相一致,A 令 $D=r$。

步骤(4):

A 发送 D 给 B。

步骤(5):

B 将 r 和 D 相比较。

C.3.2 使用 1024-bit 的 RSA 和 SHA-1 的例子

C.3.2.1 参数选择

此例使用 RSA 作为非对称加密系统的算法,GB/T 18238.3 的第 9 章中所阐明的专用散列函数 3(也可是 SHA-1)作为散列函数。

我们假设 A 使用 1024 比特的 RSA 模数 $n=pq$,一个公开 RSA 指数 e 和一个私有 RSA 指数 s,其中:

p = d329 cd1d 1156 1582 b2ec b9c3 90e8 5588 0e0e 5a6b 96b8 8e0f 8fe2 1a1c 6dd2 0d86 c85b 3932 1fdb f85d 713d aaae ad1c dab1 3571 c6d1 80a5 c0e7 7159 1be0 e4ad 54ad.

q = bbb7 9564 0e4b 12d0 6c4a bfe3 1a93 f5f4 c69e 419f eac3 3c6c 2eaf e28e a60b ad6c 81c5 7477 1092 e8b2 67c3 6176 1969 cf37 1f26 60ad e102 6c5e c1c7 f394 6fa3 f72f.

n = 9ad7 01ef 79b2 6383 a98d 4995 5bc1 3684 c32a 60c6 8690 09a7 dd06 92f8 0914 7408 83d2 183b 851d 829b bdef 4da5 a973 9e83 e9e0 bb0f 7656 05cf 878e 03c9 0cda 6456 16e4 5a3b 8da2 4bfa 5a98 a00d ba3c e534 fb9d 02ac 8408 01a9 5b27 2f05 a989 73a9 35fb 6cef 475e 64ab d367 8caf 7abc 2025 4ffb 432f dbfb a1dc 07ab 0805 25ac 76c3.

s = 8076 3fa4 ced3 052c 4e60 6a0f 4be0 336c c5c7 1a9d df78 0fd4 adc7 8493 b106 e65e 6524 7d1e de7e 1bb1 312c 2d38 aa8a 3cb2 16dd d6ed ed3b 177e 17a2 9592 82ac 4bba be1d 0e2e 0762 6e77 739e 1d70 0896 5b28 de8b 525e c1f0 a7d4 6c15 0781 a4e7 b9a3 9d2d 9ca4 2a12 8a74 aed3 35c4 9e7b 4da3 3e3b c55c e416 d27f a89f 07f1 d5f7 0423.

A 的公开加密变换为 $P_A(r)=r^e \pmod n$

C.3.2.2 鉴别交换

步骤(1):

实体 B 选择一个随机值 r。

r = fedb ad50 6bb5 2e55 e951 de0d a780 954e f6df e7a3 ac4e 859e 5dae c493 1670 afc2 84e3 37cf 3963 b13f e614 e089 77c8 2062 3ceb 2cd4 fc2f 7ecr aeaf e48a 189a e6b2 516e sb92 c4ea f516 48da e4cr 28a8 17ce 373a 40dc 9109 e255 f3e7 34d7 b1eb b03c 8a6c cb8a 2f80 4a0d 1e8a.

B 计算 $h(r)$, $r \| h(r)$ 和询问 $d = P_A(r \| h(r))$。

$$h(r) = \text{cb70 5374 99ae 42ce 1c03 0777 af95 b0a6 d978 8684.}$$

$r \| h(r)$ = fedb ad50 6bb5 2e55 e951 de0d a780 954e f6df e7a3 2c4e 859e 5dae c493 1670 afc2 84e3 37cf 3963 b13f e614 e089 77c8 2062 3ceb 2cd4 fc2f 7ec4 aeaf e48a 189a e6b2 516e 2b92 c4ea f516 48da e4cr 28a8 17ce 373a 40dc 1e8a cb70 5374 99ae 42c3 1c03 0777 af95 b0a6 d978 8684.

$d = p_A(r \| h(r))$ = 0e3d ad30 9d9e 5556 a4bd 7bab f749 a7ba c2a5 f350 fa23 07f2 72d7 8bd7 078e d5fe e00a 410c 0807 fd2c 5570 4cd3 d5db 8902 7640 0e8b 7b75 3c7c 26c1 faf4 7acc 75b0 daa7 c567 1823 d778 f432 b8a6 4457 2be2 9569 0f41 dc40 4abf e733 7b4b 2092 3d14 6ed5 9fdc d094 bd03 e5e9 bd9b 215e 7775 497b 0caf 43a7 7b85 f5eb c0eb d5b8 05cf.

步骤(2):

B 发送 d 给 A。

步骤(3):

A 执行下列计算步骤

a) A 通过以下计算恢复 r 和 $h(r)$:

$$r \| h(r) = S_A(d) = d^e \pmod n$$

b) A 根据上一步恢复的 r 值,重新计算 $h(r)$,并与在上一步恢复的 $h(r)$值相比较。

假定两个 $h(r)$的值相一致,A 置 $D=r$。

步骤(4):

A 发送 D 给 B。

步骤(5):

B 将 r 和 D 相比较。

附 录 D
（资料性附录）
机制比较

本附录对于第5章，第6章，第7章中所述的机制进行比较。

D.1 机制比较的度量

我们用下述量度标准进行比较：计算复杂性，通信复杂性，声称者认可信息的大小。

对于某些实现，声称者可以使用便携式设备（例如智能卡）来向验证者证实他的认可的合法性。在这些实现中，声称者的认可信息对智能卡的计算复杂性和通信复杂性，以及智能卡的存储能力的要求是极其重要的，因为智能卡的存储能力和处理能力同验证者的设备相比是非常有限的。如[9]所述，按1996年的科技水平，时钟频率至多为10 MHz；RAM和EEPROM容量范围分别在76到512字节和2K到20K字节。如果对本附录中所述机制的那些特点没有足够的注意，这些因素将会影响实现的效率。

因此，本附录关注由声称者所执行计算的复杂性、声称者与验证者之间的通信复杂性，以及用于保存声称者认可信息的智能卡的存储要求。

另外，在此附录中也考虑到了攻击者冒充声称者的可能性。特别地，一个不知道声称者认可信息的攻击者可能试图通过猜测询问而生成证据的方式来冒充声称者，这里计算了这种攻击成功的概率。

下列符号要使用。

C_P 计算复杂性。

C_M 通信复杂性。

S 智能卡存储大小的要求。

P 冒充的成功概率。

D.2 基于身份的机制

这里在参数值 n,v,t,m 的两种不同假设下对这个机制进行评估，它们分别对应于Fiat-Shamir方案（$v=2$）和Guillou-Quisquater方案（$v>2,t=m=1$）。

D.2.1 当 v 很大时的情形（Guillou-Quisquater 方案）

D.2.1.1 计算复杂性

在本条中，考虑 v 很大的特殊情况。特别的，当 $\lfloor \log_2 v \rfloor \geqslant 16$ 时[5]给出了一个方案。

根据定义，声称者在每一次迭代中执行下述两个计算：

$$W = r^v \bmod n$$

和

$$D = r\prod_{i=1}^{m}(C_{A_i}{}^{d_i}) \bmod n$$

本条中使用了下述符号：

N_W——计算 W 所需要的模乘法次数。

N_D——计算 D 所需要的模乘法次数。

$\mu(k)$——对于 k 比特长的模数，模乘法的计算复杂性。

大家知道（比如参见[6]），模乘法存在有效的算法使得：

$$\mu(k) = O(k^{\log_2 3})$$

下述关系成立：

$$C_p = t(N_D + N_W)\ \mu(\lfloor \log_2 n \rfloor)$$

在本条的剩余部分，我们估计 N_D 和 N_W。

利用'从右到左'形式的平方与乘法算法来计算模指数，参看[6]和[8]，计算($r^v \bmod n$)需要计算 $\lfloor \log_2 v \rfloor$ 个值：

$$r^{2^1}, \cdots\cdots, r^{2^{\lfloor \log_2 v \rfloor}}$$

那么，W 由下式决定：

$$W = (r^1)^{b_0}\ (r^2)^{b_1} \cdots (r^{2^{\lfloor \log_2 v \rfloor}})^{b_{\lfloor \log_2 v \rfloor}} \bmod n$$

其中 b_i 表示 v 的二进制表示的第 i 位比特(b_0 是最低位比特而 $b_{\lfloor \log_2 v \rfloor}$ 是最高位比特。)

对于一般的 v 来说，有一半的 $b_i(i=0,\cdots,\lfloor \log_2 v \rfloor - 1)$ 将为零，因此 N_W 的值为：

$$N_W = \frac{3}{2}\lfloor \log_2 v \rfloor$$

类似的方法用于计算 D，N_D 可由下式近似表示：

$$N_D \approx \frac{3m}{2}\lfloor \log_2 v \rfloor。$$

因此，C_P 可由下式得到：

$$C_P \approx \frac{3t}{2}(m+1)\lfloor \log_2 v \rfloor\ \mu(\lfloor \log_2 n \rfloor)$$

特别的，对于 $t=m=1$ 的情况，也就是我们熟知的 Guillou-Quisquater 方案，有：

$$C_P \approx 3\lfloor \log_2 v \rfloor\ \mu(\lfloor \log_2 n \rfloor)$$

D.2.1.2 通信复杂性

在每一轮中，声称者发送 Token$AB_1 = W$ 和 Token$AB_2 = D$ 给验证者，它们都为($\lfloor \log_2 n \rfloor$)+1 比特长。验证者也发送询问，即由 m 个比特串构成的序列，每个都为($\lfloor \log_2 v \rfloor$)+1 比特长。由于同($\lfloor \log_2 n \rfloor$)相比，m 是可以忽略的，下述式子成立：

$$C_M = t\{2(\lfloor \log_2 n \rfloor + 1) + m(\lfloor \log_2 v \rfloor + 1)\}$$
$$\approx t(2\lfloor \log_2 n \rfloor + m\lfloor \log_2 v \rfloor)$$

在 Guillou-Quisquater 方案中，我们有：

$$C_M \approx 2\lfloor \log_2 n \rfloor + \lfloor \log_2 v \rfloor$$

声称者也允许发送 Token$AB_1 = h(W \parallel \text{Text})$以代替 Token$AB_1 = W$。在这种情况下，$C_M$ 为：

$$C_M \approx t(\lfloor \log_2 n \rfloor + H + m\lfloor \log_2 v \rfloor)$$

这里 H 是散列函数 h 的散列码的比特数。

D.2.1.3 声称者的认可大小

除了公开密钥对(v,n)之外，声称者保留私有认可信息 C_{A_1}，C_{A_2}，$\cdots\cdots C_{A_m}$。因此，所需要的存储比特为：

$$S \approx (m+1)\lfloor \log_2 n \rfloor + \lfloor \log_2 v \rfloor。$$

在 Guillou-Quisquater 方案中，我们有：

$$S \approx 2\lfloor \log_2 n \rfloor + \lfloor \log_2 v \rfloor$$

D.2.1.4 安全程度

一个不知道声称者认可信息的攻击者，可能通过猜测验证者将要发送的询问，然后生成证据的方法，试图冒充声称者。攻击者冒充声称者的成功概率为：

$$P = \frac{1}{v^m}$$

对于 Guillou-Quisquater 方案，我们有：

$$P = \frac{1}{v}$$

D.2.2 Fiat-Shamir 方案

在本条中，考虑 $v=2$ 的特殊情况。在这种情况下，此机制被称为 Fiat-Shamir 方案。

D.2.2.1 计算的复杂性

由于 $v=2$，我们有 $\lfloor \log_2 v \rfloor = 1$，因此

$$N_W = 1, N_D = \frac{m}{2}$$

所以，

$$C_P = \frac{1}{2} t(m+2)\mu(\lfloor \log_2 n \rfloor)$$

D.2.2.2 通信复杂性

由于同 $\lfloor \log_2 n \rfloor$ 相比较 m 是可忽略的，有

$$C_M \approx 2t \lfloor \log_2 n \rfloor$$

声称者也允许发送 Token$AB_1 = h(W \parallel \text{Text})$代替 Token$AB_1 = W$。在这种情况下，$C_M$ 为：

$$C_M \approx t(\lfloor \log_2 n \rfloor + H)$$

其中 H 是散列函数 h 的散列码的比特数。

D.2.2.3 声称者的认可大小

$$S \approx (m+1)\lfloor \log_2 n \rfloor。$$

D.2.2.4 安全程度

攻击者成功地猜测所有 mt 个询问的概率为：

$$P = \frac{1}{2^{mt}}$$

D.3 使用离散对数的基于证书的机制

D.3.1 计算复杂性

声称者执行的计算是：

$$W = g^r \bmod p$$

和

$$D = r - dz_A \bmod q$$

由于计算 D 的复杂性与计算 W 相比较是可以忽略的，那么整个复杂性 C_P 为：

$$C_P \approx \frac{3}{2}\lfloor \log_2 q \rfloor \mu(\lfloor \log_2 p \rfloor)$$

D.3.2 通信复杂性

声称者发送 Token$AB_1 = W$ 和 Token$AB_2 = D$ 给验证者，它们分别为 $(\lfloor \log_2 p \rfloor + 1)$ 和 $(\lfloor \log_2 q \rfloor + 1)$ 比特长。验证者发送一个询问，长为 $(\lfloor \log_2 q \rfloor + 1)$ 比特，因此：

$$C_M \approx \lfloor \log_2 p \rfloor + 2\lfloor \log_2 q \rfloor$$

声称者也允许发送 Token$AB_1 = h(W \parallel \text{Text})$以代替 Token$AB_1 = W$。在这种情况下，我们有：

$$C_M \approx H + 2\lfloor \log_2 q \rfloor$$

其中 H 是散列函数 h 的散列码的比特数。

D.3.3 声称者的认可大小

除了三个正整数 p，q 和 g 之外，声称者需要存储私有密钥 z_A，因此：

$$S \approx 2(\lfloor \log_2 p \rfloor + \lfloor \log_2 q \rfloor)$$

D.3.4 安全程度

询问 d 选自集合 $\{0, 1, \ldots\ldots, q-1\}$。因此，攻击者猜测一个询问成功的概率为：

$$P=\frac{1}{q}$$

D.4 使用非对称加密系统的基于证书的机制

为了本章的讨论，假设 RSA 密码系统是机制的非对称加密系统。下述符号要用到：

a) (y_A, n)是声称者的公开密钥。

b) z_A 是声称者的私有密钥，也就是说，关系式 $y_A z_A \equiv 1 (\mathrm{mod}\ \lambda(n))$成立。

c) $P_A(m)=m^{y_A} \mathrm{mod}\ n$.

d) $S_A(c)=C^{z_A} \mathrm{mod}\ n$.

D.4.1 计算复杂性

声称者执行的计算为：

$$S_A(d)=d^{z_A} \mathrm{mod}\ n$$

因此，计算复杂性为：

$$C_P \approx \frac{3}{2} \lfloor \log_2 n \rfloor \mu(\lfloor \log_2 n \rfloor)$$

D.4.2 通信复杂性

验证者发送$(\lfloor \log_2 n \rfloor+1)$比特长的询问 $P_A(r \| h(r))$，而声称者发送 Token$AB=r$ 返回给验证者。假设$\lfloor \log_2 (r \| h(r)) \rfloor \approx \lfloor \log_2 n \rfloor$，Token$AB$ 的比特长度是$\lfloor \log_2 n \rfloor - \lfloor \log_2 h(r) \rfloor$。因此：

$$C_M \approx 2 \lfloor \log_2 n \rfloor - H$$

其中 H 是散列函数 h 的散列码的比特数。

D.4.3 声称者的认可大小

声称者保存的认可信息，是 RSA 私有密钥对(z_A, n)，它的长度用比特表示是：

$$S \approx 2 \lfloor \log_2 n \rfloor$$

D.4.4 安全程度

攻击者成功地猜测到 r 值的概率为：

$$P=\frac{1}{2^{\lfloor \log_2 n \rfloor - H}}$$

D.5 机制的比较

表 D.1 概括了在第 D.2～第 D.4 章中讨论的参数。

另一方面，表 D.2 给出了当参数选定为某些具体值时这些机制的比较(例如：$\lfloor \log_2 n \rfloor$，$\lfloor \log_2 v \rfloor$，$\lfloor \log_2 p \rfloor$)。表 D.2 中的每一个值都是其中一个机制的计算值与 Fiat-Shamir 方案的相应计算值的比率。

我们采用下面具体的参数选取来获得表 D.2 中提供的数字。

——Fiat-Shamir 方案(*FS*)。假定值 $m=2$ 和 $t=10$，作为 Fiat 和 Shamir 的参数，[3]，(推荐)。

——带散列承诺的 Fiat-Shamir 方案(*FSH*)。这是 Token$AB_1=h(W \| \mathrm{Text})$的情形。由于许多现有的散列函数都产生 128-比特的散列值，我们把 H 置为 128。其他参数值与上面的 Fiat-Shamir 方案相同。

——Guillou-Quisquater 方案(*GQ*)。假定值$\lfloor \log_2 v \rfloor=16$。

——Schnorr 方案(*SC*)。Schnorr[10]推荐 $\log_2 p=512$ 和 $\log_2 q=140$。

——带散列承诺的 Guillou-Quisquater 方案(*GQH*)。这是 Token$AB_1=h(W \| \mathrm{Text})$的情形。由于许多现有的散列函数都产生 128-bit 的散列值，我们把 H 定为 128。其他参数值与上面的 GQ 方案一样。

——使用非对称加密系统(*RSA*)的基于证书的机制。由于许多现有的散列函数都产生的 128-比

特的散列值，我们把 H 定为 128。

表 D.1 评估函数

	C_P	S	C_M	P
FS	$(1/2)t(m+2)\mu(\lfloor \log_2 n \rfloor)$	$(m+1)\lfloor \log_2 n \rfloor$	$2t\lfloor \log_2 n \rfloor$	$1/2^{m}$
FSH	$(1/2)t(m+2)\mu(\lfloor \log_2 n \rfloor)$	$(m+1)\lfloor \log_2 n \rfloor$	$t(\lfloor \log_2 n \rfloor+H)$	$1/2^{m}$
GQ	$3\lfloor \log_2 v \rfloor\mu(\lfloor \log_2 n \rfloor)$	$2\lfloor \log_2 n \rfloor+\lfloor \log_2 v \rfloor$	$2\lfloor \log_2 n \rfloor+\lfloor \log_2 v \rfloor$	$1/v$
GQH	$3\lfloor \log_2 v \rfloor\mu(\lfloor \log_2 n \rfloor)$	$2\lfloor \log_2 n \rfloor+\lfloor \log_2 v \rfloor$	$\lfloor \log_2 n \rfloor+H+\lfloor \log_2 v \rfloor$	$1/v$
SC	$\frac{3}{2}\lfloor \log_2 q \rfloor\mu(\lfloor \log_2 p \rfloor)$	$2(\lfloor \log_2 p \rfloor+\lfloor \log_2 q \rfloor)$	$\lfloor \log_2 p \rfloor+2\lfloor \log_2 q \rfloor$	$1/q$
SCH	$\frac{3}{2}\lfloor \log_2 q \rfloor\mu(\lfloor \log_2 p \rfloor)$	$2(\lfloor \log_2 p \rfloor+\lfloor \log_2 q \rfloor)$	$H+2\lfloor \log_2 q \rfloor$	$1/q$
RSA	$\frac{3}{2}\lfloor \log_2 n \rfloor\mu(\lfloor \log_2 n \rfloor)$	$2\lfloor \log_2 n \rfloor$	$2\lfloor \log_2 n \rfloor-H$	$2^{H-\lfloor \log_2 n \rfloor}$

表 D.2 特殊参数选择的评估比率

	C_P	S	C_M	P
FS	1	1	1	1
FSH	1	1	0.625	1
GQ	3	0.67969	0.10195	1
GQH	3	0.67969	0.0644531	1
SC	10.5	0.75781	0.77344	2^{-120}
SCH	10.5	0.75781	0.0398437	2^{-120}
RSA	38.4	0.66667	0.0875	2^{-354}

附 录 E
（资料性附录）
关于专利的信息

GB/T 15843 的本部分在制定中等同采用了国际标准 ISO/IEC 9798-5:1999 附录 E 的专利信息。

关于专利的使用遵照国家有关规定。

国际标准的有关专利信息如下：

在 GB/T 15843 本部分的准备期间，收集到了 ISO/IEC 9798 本部分的应用可能涉及到的有关专利信息。有关专利信息如下表所示。但是，关于专利的有效性或范围，ISO/IEC 不能够提供官方的或全面的信息。

这些注册专利的持有者已经声明，如果寻求授权许可的使用方同意付费，将在适当的条件下给予许可，使其能够应用 GB/T 15843 的本部分。

更多的信息请向相关的专利持有者询问。

专业范围	发明者	专利号	发布日期	联系地址
Fiat-Shamir 鉴别	Shamir-Fiat	US 4,748,668	1988-05-31	News Difital Systems Ltd. Stoneham Rectory Stoneham Lane Eastleigh Hampshire SO50 9NW, UK
Schnorr 签名	Schnorr	US 4,995,082	1991-02-19	RSA Data Security Inc. Director of Licensing 2955 Campus Drive. Suite 400 San Mateo,CA 94403-2507 USA
GQ 鉴别	Guillou-quisquator	US 5,140,634	1992-08-18	CCETT Patent and IPR Office BP 59 4 Rue du Clos Courtel F-35512 Cesson Sevigne France
		EP 0.311.470	1992-12-16	Philips International B. V. Corporate Patents and Trademarks P. O. Box 220 56000 AE Eindhoven The Netherlands

附 录 F
（资料性附录）
参考文献

[1] J. Brandt, I. Damgad, P. Landock and T. Pedersen, Zero-knowledge authentication scheme with sectet key exchange, in Advances in Cryptology -CRYPTO '88', S. Goldwasser(editor), Springer-Verlag, Berlin (1990), pp. 583-588

[2] C. Feige, A. Fiat and A. Shamir, Zero knowledge proofs of identity journal of Cryptology, Volume 1 (1988) pp. 77-94

[3] A. Fiat and A. Shamir, How to prove yourself: Practical solutions to identificationand signature problems, in: 'Advances in Cryptology-CRYPTO '86', A. M. Odlyzko (editor), Springer-Verlag, Berlin (1987), pp. 186-194'

[4] S. Goldwasser, S. Micali and C. Rackoff, The knowledge complexity of interactive proof systems. SIAM Joural on Computing. Volume 18(1989)pp. 186-208

[5] L. C. Guillou and J. -J. Quisquater. A practical zero-knowledge protocal fitted to security microprocessor minimizing both transmission and memory in: 'Advances in Cryptology-EUROCRYPT '88', C. G. Gunther(editor) Springer-Verlag. Berlin (1988). pp. 123-128

[6] D. E. Knuth, The art of Computer Programming, Volume 2. Addison-Wesley (2^{nd} edition, 1981)

[7] C. H. Lim and P. J. Lee. A key recovery attack on discrete log based schemes using a prime order subgroup, in: 'Advances in Cryptology-Crypto' 07', Lecture Notes in Computer Science 1294, Springer Verlag. Berlin (1997). pp. 249-263'

[8] A. J. Menezes. . P. C van Oorschot and S. A. Vanstone, Handbook of applied cryptography. CRC Press. Boca Raton (1997)

[9] D. Naccache. D. M'Raihi. Cryptographic smart cards. IEEE Micro. Volume 16 no. 3 (June 1996)

[10] C. P. Schnorr. Efficient iddentification and signatures for smart cards. In: 'Advances in Cryptology-CRYPTO '89'. G. Brassard (editor), Springer-Verlag. Berlin (1990). pp. 239-252