



中华人民共和国国家标准

GB/T 17902.1—1999

信息技术 安全技术 带附录的数字签名 第1部分:概述

Information technology—Security techniques—
Digital signature with appendix—
Part 1: General

1999-11-11 发布

2000-05-01 实施

国家质量技术监督局 发布

前 言

本标准规定了带附录的数字签名方案,适合于我国使用。

GB/T 17902 在总标题《信息技术 安全技术 带附录的数字签名》下,由以下几个部分组成:

第 1 部分:概述;

第 2 部分:基于身份的方案;

第 3 部分:基于证书的方案。

本标准的附录 **A** 是标准的附录,附录 **B** 是提示的附录。

本标准由国家信息化办公室提出。

本标准由全国信息技术标准化技术委员会归口。

本标准由复旦大学、中科院软件所负责起草。

本标准主要起草人:鲍振东、赵一鸣、陶仁骥、计青。

引 言

数字签名机制采用非对称密码技术,它可用来提供实体鉴别,数据原发鉴别,数据完整性和抗抵赖服务。有两种数字签名机制:

——若验证进程需要消息作为输入部分,这种机制称为“带附录的数字签名”。在计算附录时使用了散列函数。**ISO/IEC 10118** 规定了这类散列函数;

——若验证进程给出消息及其特定冗余(有时也称作消息影子),这种机制称为“带消息恢复的签名机制”。**GB 15851** 规定了这种机制。

这两种机制不是互斥的。具体地说,任何带消息恢复的签名机制,例如,**ISO/IEC 9796-1** 规定的机制,可以用来提供带附录的数字签名。这种情况下,可以对消息的散列权标使用签名进程来产生签名。

信息技术 安全技术
带附录的数字签名
第1部分:概述

GB/T 17902.1—1999

Information technology—Security techniques—
Digital signature with appendix—
Part 1:General

1 范围

系列标准 GB/T 17902 规定了几个任意长度消息的带附录数字签名机制。本标准包括了带附录的数字签名的基本原则和要求,同时也包括了在该系列标准的所有部分都用到的定义和符号。

2 引用标准

下列标准所包含的条文,通过在本标准中引用而构成为本标准的条文。本标准出版时,所示版本均为有效。所有标准都会被修订,使用本标准的各方应探讨使用下列标准最新版本的可能性。

ISO/IEC 9796-1:1991 信息技术 安全技术 带消息恢复的数字签名方案 第1部分:使用冗余的机制

ISO/IEC 9796-2:1997 信息技术 安全技术 带消息恢复的数字签名方案 第2部分:使用散列函数的机制

ISO/IEC 10118-1:1996 信息技术 安全技术 散列函数 第1部分:概述

ISO/IEC 11770-3¹⁾ 信息技术 安全技术 密码管理 第3部分:使用非对称密码技术的机制

3 概述

本标准所规定的机制是基于非对称密码技术的。所有非对称数字签名机制涉及三个基本操作:

- 产生密钥对的进程:每对密钥包括签名密钥和相应的验证密钥;
- 使用签名密钥的进程:称为签名进程;
- 使用验证密钥的进程:称为验证进程。

数字签名的验证需要签名实体的验证密钥。所以,验证方必须把正确的验证密钥与签名实体,或者更准确地讲,与(部分)签名实体的标识数据联系起来。如果这种联系是验证密钥自身所固有的,这种方案是“基于身份的”。如果不是,应该由其他途径提供正确的验证密钥与签名实体间的联系,无论使用何种途径,这种方案是“基于证书的”。

基于证书方案的验证密钥管理超出本标准的范围。ISO/IEC 11770-3 提供了公开密钥分发的机制。

1) 待发布。

4 术语和定义

4.1 附录 **appendix**

由签名和一个任选文本字段构成的位串。

4.2 赋值 **assignment**

赋值是一个数据项,它是消息的证据函数或可能是部分消息的证据函数,赋值形成签名函数的部分输入。

4.3 无碰撞散列函数 **collision resistant hash-function**

满足以下性质的散列函数:

要找到两个不同的输入使之对应同一输出,在计算上是不可行的。

注:计算上的可行性依赖于用户的具体安全要求和环境。

4.4 确定性的 **deterministic**

与随机值无关,非随机化的。

4.5 数字签名 **digital signature**

参见“签名”。

4.6 域参数 **domain parameter**

一个在域中所有实体公共的且已知或可访问的数据项。

4.7 散列代码 **hash-code**

一个散列函数的输出位串。

4.8 散列函数 **hash-function**

将位串映射成固定长度位串的函数,满足以下两个性质:

对于一个给定的输出,要找到其对应的输入,在计算上是不可行的;

对于一个给定的输入,要找到对于其输出的第二个输入,在计算上是不可行的。

注:计算上的可行性依赖于用户的具体安全要求和环境。

4.9 散列权标 **hash-token**

散列代码与一个用来识别散列函数和填充方法的任选控制字段的拼接。

4.10 标识数据 **identification data**

分配给实体以便标识它的数据项序列,包括实体的区分标识符。

注:标识数据可以额外包含诸如签名进程的标识符,签名密钥的标识符,签名密钥的有效周期,密钥使用的约束,相关的安全策略参数,密钥序号,或域参数等数据项。

4.11 消息 **message**

一个任意长度的位串。

4.12 预签名 **pre-signature**

在签名进程中计算出来的值,它是随机值的函数,并且与消息无关。

4.13 随机化 **randomized**

依赖于随机值。

4.14 随机值 **randomizer**

预签名进程中签名实体产生的一个秘密的、不可预测的值。

4.15 签名 **signature**

签名进程输出的位串。

注:这个位串可能就有签名机制的特定内部结构。

4.16 签名方程 **signature equation**

一个签名函数的形式。

4.17 签名函数 signature function

签名进程中,由签名密钥和域参数决定的函数。

一个签名函数把赋值和可能的随机值作为输入,并给出签名的第二部分作为输出。

4.18 签名密钥 signature key

签名进程中,实体所特有的并只能由这个实体所使用的秘密数据项。

4.19 签名进程 signature process

将消息、签名密钥和域参数作为输入,并输出签名的进程。

4.20 已签名消息 signed message

一组包含签名、不能从签名恢复的部分消息和任选文本字段所组成的数据项。

注:在本标准中,整个消息包含在已签名消息中,并且消息的任何部分都不能从签名中得到恢复。

4.21 验证函数 verification function

验证进程中,一个由验证密钥决定,并给出一个重新计算的证据值作为输出的函数。

4.22 验证密钥 verification key

一个与实体的签名密钥有关,在验证进程中由验证方使用的数据项。

4.23 验证进程 verification process

一个把已签名消息、验证密钥和域参数作为输入,把验证签名的结果,即有效或无效作为输出的进程。

4.24 证据 witness

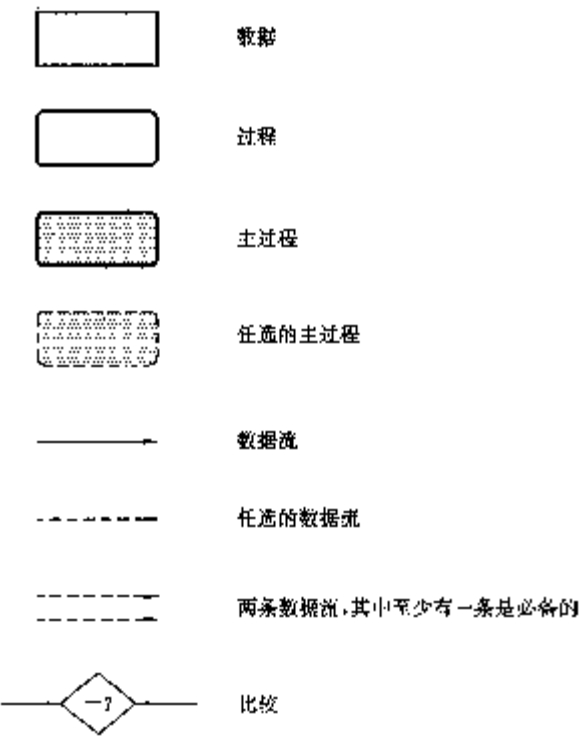
一个为验证方提供证明的数据项。

5 符号和图中使用的图例

在 GB/T 17902 中,将使用以下符号:

H	散列权标
\bar{H}	重新计算的散列权标
K	随机值
M	消息
M_1, M_2	准备好的部分消息
R	签名的第一部分
\bar{R}	重新计算的签名的第一部分
S	签名的第二部分
T	赋值
X	签名密钥
Y	验证密钥
Z	一个或多个域参数的集合
Π	预签名
$\bar{\Pi}$	重新计算的预签名
Σ	签名
$A \bmod N$	整数 A 除以整数 N 后得到的余数
$A \equiv B \pmod{N}$	整数 A 与整数 B 模 N 同余,即 $(A-B) \bmod N = 0$

以下是 GB/T 17902 的图中的图例。



6 一般模型

- 一个带附录的数字签名机制由以下进程定义：
- 密钥产生进程；
 - 签名进程；
 - 验证进程。

在签名进程中，签名实体对给定的消息计算其数字签名。这个签名和一个任选的文本字段形成附录，用来附加在消息上形成已签名消息。

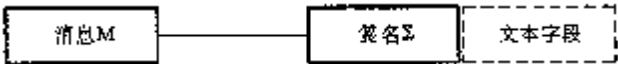


图 1 已签名消息

根据应用由不同方法形成附录并附加在消息上。基本要求是验证方可以将正确的签名和消息关联起来。

给签名进程输入的消息可能是一个待鉴别的原始消息的散列权标。在这种情况下，附录附加在原始消息上，散列函数标识符形成附录中文本字段的必备部分。

为了成功验证，还要求在验证进程之前，验证方能够得到该签名的正确验证密钥。任选的文本字段可以用来向验证方传送签名方的标识数据，或签名方的验证密钥的可鉴别拷贝。在某些情况下，签名方的标识数据可能应是消息 **M** 的一部分，这样它可以受到签名的保护。

一个数字签名机制应该满足以下要求：

- 只给出验证密钥，不知道签名密钥，想产生任何消息和该消息的有效签名在计算上是不可行的；
- 签名方产生的签名，不可以用来产生任何新的消息和该消息的有效签名，也不可以用来恢复签名密钥；
- 即使对于签名方而言，想找到具有相同签名的两个不同消息，在计算上是不可行的。

注：计算上的可行性依赖于用户的具体安全要求和环境。

7 密钥产生进程

数字签名机制的密钥产生进程由以下两个过程组成：

- 产生域参数；
- 产生签名密钥和验证密钥。

在建立域时，第一个过程运行一次。域参数的结果集 **Z** 要用在后继的进程和函数中。对于域中的每个签名实体，第二个过程都要运行，其输出就是签名密钥 **X** 和验证密钥 **Y**。对于域参数中一个具体的集合，将要用的值 **X** 应以很高的概率不同于以前使用过的值。

8 签名进程

签名进程需要以下数据项：

- 域参数 **Z**；
- 签名密钥 **X**；
- 消息 **M**；
- 散列函数标识符(任选的)；
- 其他文本(任选的)。

散列函数标识符可以用来绑定签名机制和散列函数，见附录 A。

带附录的数字签名机制的签名进程由以下过程组成：

- 产生预签名；
- 为签名准备消息；
- 计算证据；
- 计算签名。

第一个过程是任选的。一个没有预签名的签名机制称为确定性的。一个有预签名的签名机制称为随机化的。

数字签名的证据是一个数据项，它的值在签名进程中确定。证据的正确性在验证进程中验证。证据作为消息的函数或预签名函数或两者的函数，可以计算出来。

证据与预签名无关或预签名不存在时，称它是确定性的。一个确定性的证据不必给予验证方，验证方可以使用和签名方同样的方法，将证据作为消息的一个函数来计算。图 2 描述了带确定性证据的签名进程。

如果证据依赖于预签名，称它为随机化的。随机化证据的值由签名方计算出来，并形成签名的第一部分。图 3 描述了带随机化证据的签名进程。

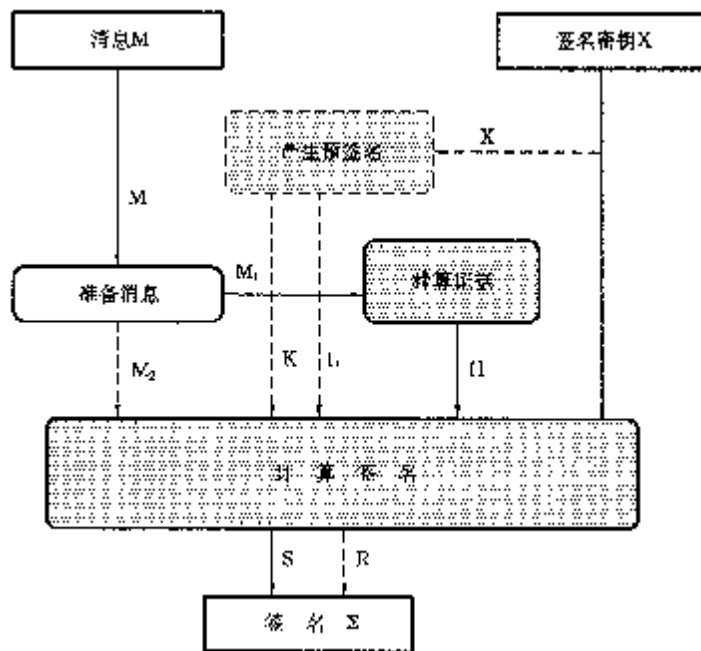


图2 带确定性证据的签名进程

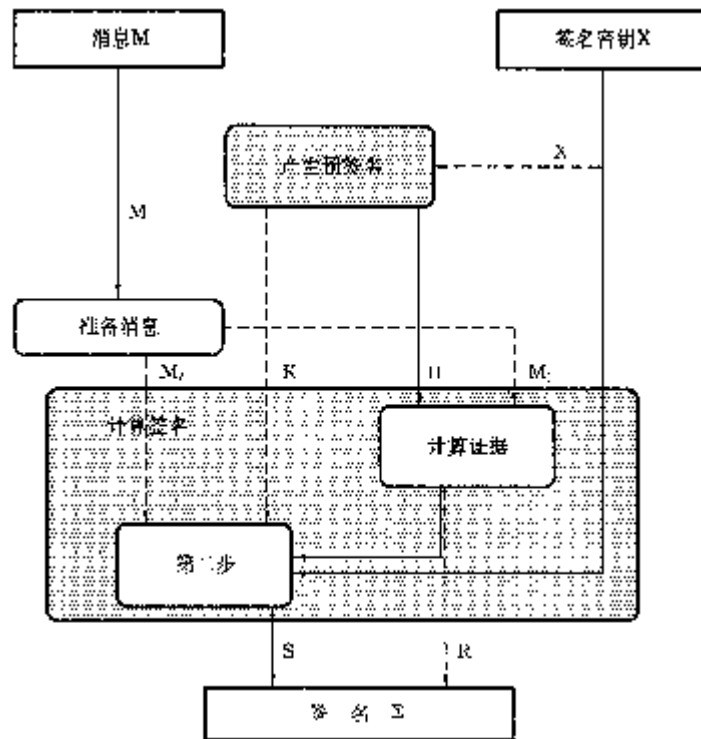


图3 带随机化证据的签名进程

8.1 产生预签名

预签名过程是随机化签名机制所需要的,它由以下两步组成:

- 产生随机值 K ;
- 计算预签名 Π 。

第一步的输出是随机值 K ,它是一个仅可在签名进程中使用的秘密值。对于每个消息,(在签名密钥

的生命周期内)以很高的概率不同于以前使用过的值 K 将用来保护签名密钥的秘密性(见附录 B 的 [1])。第二步中,通过使用由域参数 Z 、还可能由签名密钥 X 所共同决定的函数,由 K 计算得到预签名 Π 。其输出是随机值 K 和预签名 Π 。

预签名的计算是独立于消息的。因此,可以脱机产生随机值和计算相应的预签名并安全地储存起来,以便今后在签名进程中使用。如果随机值是作为消息和签名密钥的(伪随机)函数计算得到,那么随机化机制的这个特征就不能利用。

8.2 准备消息

签名进程可以将(部分)消息作为计算证据或计算(第二部分的)签名或两者的输入。为此, M_1 和 M_2 这两个数据字段就可由消息 M 导出。准备消息的进程应当满足以下两个条件中的一个:

- 给定 M_1 和 M_2 ,可以重构整个消息 M ;
- 要找到两个消息 M 和 M' ,使得它们的两个导出对 (M_1, M_2) 与 (M'_1, M'_2) 相等,在计算上是不可行的。

特别地,在第一种情况中, $M_1=M, M_2$ 为空;或 $M_2=M, M_1$ 为空;或 $M_1=M_2=M$ 。在第二种情况中, M_1 或 M_2 或两者是 M 的散列权标。

8.3 计算证据

确定性证据是用无碰撞散列函数(见图 2)计算出来的 M_1 的散列权标 H 。

随机化证据依赖于预签名 Π 和 M_1 ,后者可任选。随机化证据是作为签名计算的一部分而求得的,见 8.4 的描述。

8.4 计算签名

在确定性机制中,过程的输入是证据 H 、签名密钥 X 和部分消息 M_2 ,这里 M_2 可任选。这种情况下,这一步的输出 S 是签名 Σ ,见图 2。

在带确定性证据的随机化机制中,这个过程的输入是随机值 K 、签名密钥 X 、确定性证据 H 和预签名 Π 。过程的输出是整个签名 Σ ,它或者由 S 组成,或者由 R 和 S 两部分组成,见图 2。

在带随机性证据的随机化机制中,这个过程由两步组成。第一步,计算证据 R 。证据 R 依赖于预签名 Π 和 M_1 , M_1 是任选的。如果计算证据时使用了散列函数,那么可能需要将散列函数的标识符和散列函数的输出拼接起来(见附录 A)。在第二步,它的输入是随机值 K 、签名密钥 X 、随机性证据 R 和已准备的部分消息 M_2 , M_2 是任选的,第二步的输出是 S 。签名 Σ 或者由 S 组成,或者由 R 和 S 两部分组成,见图 3。

9 验证进程

验证进程需要以下数据项:

- 域参数 Z ;
- 验证密钥 Y ;
- 消息 M ;
- 签名 Σ ;
- 散列函数标识符(任选的);
- 其他文本(任选的)。

带附录的数字签名机制的验证进程由以下过程组成:

- 为验证准备消息;
- 检索证据;
- 计算验证函数;
- 验证证据。

为了确定正确的验证密钥,为验证准备消息可能涉及从消息 M 中提取签名实体的标识数据。

如果证据是确定性的,验证方将证据的值作为消息的函数来检索。图4描述了这个验证进程。其他情况见图5,验证方从签名中检索证据。第一个验证进程的好处是证据的检索与重新计算可以并行进行。

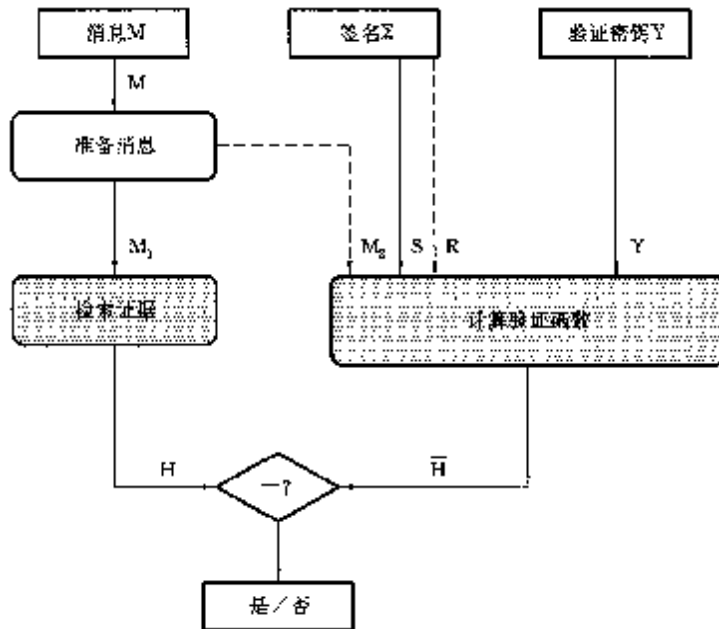


图4 带确定性证据的验证进程

9.1 准备消息

这个过程应该等同于 8.2。它的输入是消息 M , 输出是消息 M_1 和 M_2 两部分。

9.2 检索证据

确定性证据 H 是通过使用与 8.3 中签名方一样的无碰撞散列函数计算消息 M_1 的散列权标而检索得到的。

若不是确定性证据,验证方将从签名 Σ 的第一部分或验证函数的输出中检索得到随机化证据 R (见图5)。

9.3 计算验证函数

验证函数由签名方的公开密钥 Y 决定。签名 Σ 是验证函数的必备输入。如果证据是随机化的,消息 M 的非空部分 M_1 、 M_2 作为这步的必备输入(见图5)。如果证据是确定性的,只有 M_2 是验证函数的输入(见图4)。

这个过程的输出是证据的重新计算值,即 \bar{H} 或 \bar{R} 。如果随机化证据 R 不是签名的部分,那么它作为这个过程的第二个输出,由验证方恢复,见图5。

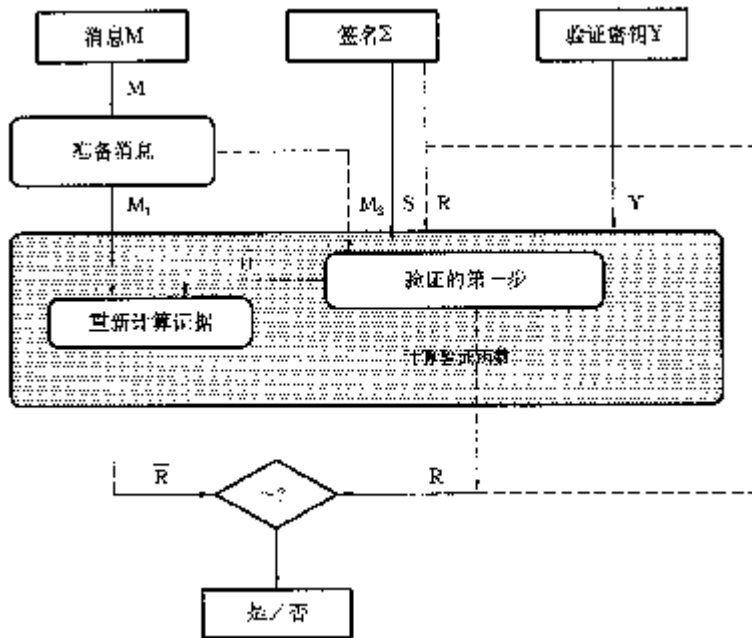


图 5 带随机化证据的验证进程

9.4 验证证据

这一步中,要比较两个证据的值,一个是用 9.2 的方法检索的,另一个是用 9.3 的方法重新计算的。如果这两个值相等,那么验证方就证明了消息 M 的签名 Σ 是用与验证密钥 Y 所对应的签名密钥 X 获得的。

10 带两部分签名的随机化机制

这一章是对第 8 章和第 9 章所描述模型的改进。这些改进适用于由两部分计算得到签名的随机化数字签名机制。

10.1 计算签名

在随机化签名机制中,计算签名需要以下数据项:

- 域参数 Z ;
- 签名密钥 X ;
- 消息的第二部分 M_2 ;
- 证据 H 或 R ;
- 随机值 K ;
- 预签名 Π (如果证据是确定性的,它是必备的);
- 散列函数标识符(任选的)。

在随机化签名机制中,计算两部分签名需要以下步骤:

- 计算签名的第一部分;
- 计算赋值;
- 计算签名的第二部分。

对于带确定性证据的机制,图 6 描述了这些步骤,它是就随机化机制的特殊情况,即对图 2 中“计算签名”框的放大。

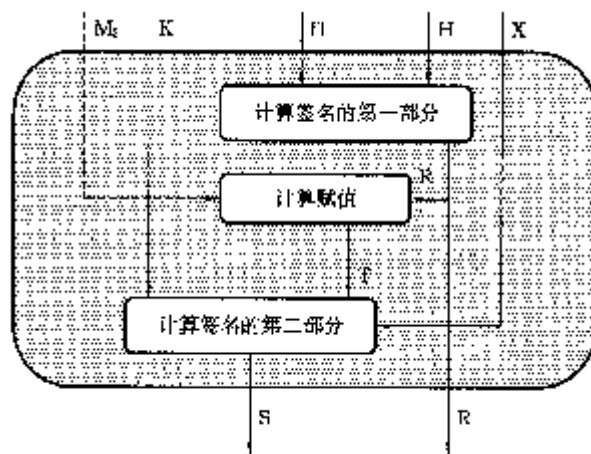


图6 带确定性证据的随机化机制中签名的计算

如果证据是随机化的,它形成签名的第一部分。图7描述了计算签名所需的步骤,它是图3中“计算签名”框的放大。

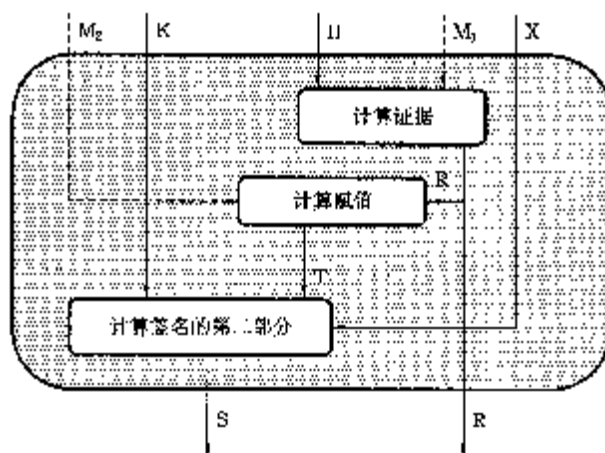


图7 把随机化证据作为签名一部分的签名计算

要求对于所有的预签名,要找到两个消息有相同的证据和赋值,在计算上是不可能的。在多数随机化数字签名机制中,或 $M_1=M$, M_2 为空;或 M_1 为空, $M_2=M$ 。第一种情况下,证据 H 或 R 用无碰撞散列函数计算得到。在第二种情况, M_1 为空,证据仅依赖于预签名,赋值 T 用无碰撞散列函数计算得到。除非散列函数由签名机制或域参数唯一确定,散列函数的标识符必须包含在赋值中。

10.1.1 计算签名的第一部分

这一步的输入是证据 H 和预签名 Π ,输出是签名的第一部分 R 。

这一步的计算按以下方式应该是可逆的:给定 R 和 Π ,验证方应该可以计算出 H 。

10.1.2 计算赋值

这一步的输入是签名的第一部分 R ,也可能是部分消息 M_2 ;输出是赋值 T 。

10.1.3 计算签名的第二部分

签名函数由签名密钥 X 决定。把随机值 K 和赋值 T 作为输入,签名的第二部分 S 作为输出。

在某些签名机制中,签名函数可由一个方程式给出,该方程式由参数 X 、 K 和 T 以及要求解的未知参数 S 所确定。

10.2 计算验证函数

计算验证函数需要以下数据项:

- 域参数 Z ;
- 验证密钥 Y ;

- 签名 $\Sigma=(R,S)$;
- 消息的第二部分 M_2 (任选的);
- 散列函数标识符(任选的)。

随机化签名机制中的验证计算由以下步骤组成:

- 检索赋值;
- 重新计算预签名;
- 重新计算证据。

对于带确定性证据的随机化机制,图8描述了这些步骤,它是就随机化机制的特殊情况,对图4中相应框的放大。如果证据是随机化的,图9描述了它的重新计算,它是图5中相应框的细分。

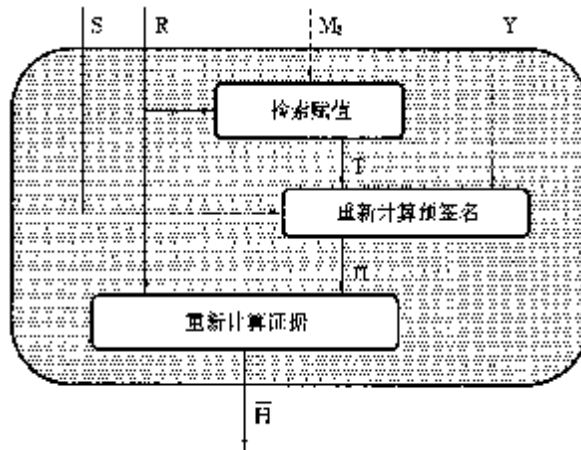


图8 带确定性证据的随机化机制中验证函数的计算

10.2.1 检索赋值

这一步等同于 10.1.2, 它的输出是赋值 T 。

10.2.2 重新计算预签名

这一步由验证密钥 Y 决定。这一步的其他输入是签名的第二部分 S 和 10.2.1 中得到的赋值 T 。它的输出是为预签名重新计算的值 $\bar{\Pi}$ 。

10.2.3 重新计算证据

如果证据是确定性的,那么按照 10.1.1 的条件,验证方可以使用预签名重新计算的值 $\bar{\Pi}$ 和签名的第一部分 R 决定重新计算值 \bar{H} (见图8)。

如果证据是随机化的,那么验证方可以使用和 8.3 中签名实体一样的计算过程,由重新计算的预签名 $\bar{\Pi}$ 和消息的第一部分 M_1 (任选的)决定一个重新计算的证据值 \bar{R} (见图9)。

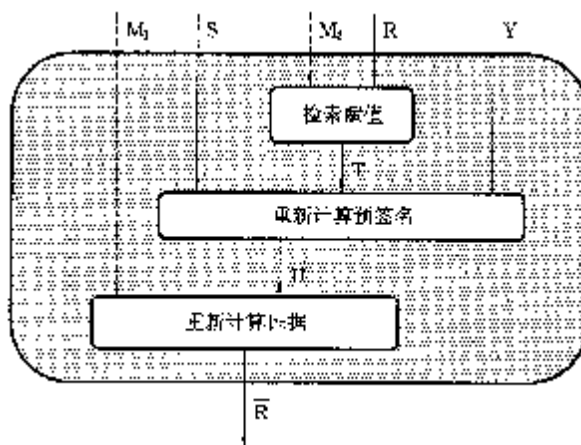


图9 把随机化证据作为部分签名的验证函数计算

附 录 A

(标准的附录)

绑定签名机制和散列函数的安全性注释

当数字签名机制使用散列函数时,签名机制和散列函数之间应当绑定。如果没有这样的绑定,入侵者可能会声明使用了一个弱散列函数(并不是真正的那个),然后伪造一个签名。有很多方法可以用来完成所需的绑定。以下选择按风险程度由低到高列出。

1 当使用一个特定的签名机制时,需要一个特定的散列函数。验证进程应当唯一地使用该特定的散列函数。GB/T 17902 的第 3 部分将给出这种选择的一个例子,其中 DSA 机制需要使用 SHA-1。

2 允许有一个散列函数的集合,并在每次签名中用包含在部分签名计算的散列函数标识符来清楚地表示出所使用的散列函数。散列函数标识符是散列代码的扩展:它表示如何导出散列代码。验证进程应当唯一地使用签名过程中表示的散列函数。ISO/IEC 9796 的第 2 部分给出了这种选择的一个例子。

3 允许有一个散列函数的集合,并在证书域参数中清楚地表示出所使用的散列函数。在证书域的内部,验证进程应当唯一地使用证书所示的散列函数。在证书域外部,则存在由非正规认证机构带来的风险。如果可以创建其他证书,那么就可以创建其他签名。受到攻击的用户就可能与其他证书的认证机构存在争议。

4 允许有一个散列函数的集合,并用其他方法指明所使用的散列函数,如消息或双边协议中指明的散列函数。验证进程应当唯一地使用那种方法指明的散列函数。但是,这里有一个风险,就是入侵者可能会使用另一个散列函数来伪造签名。

数字签名机制的用户应当对各种选择的开销和好处进行风险评估。这个评估包括与产生伪造签名的可能性有关的开销。

附 录 B

(提示的附录)

参 考 文 献

[1] T. Elgamal, 基于离散算法的公开密钥密码体制和签名机制, IEEE 会刊 JT-31(4), 1985, 469-472