一种获取数字签名和公钥密码系统的方法

R.L. Rivest, A. Shamir, 和 L. Adleman*

摘要

本文提出了一种新颖的加密方法，其特点是公开加密密钥并不会泄露相应的解密密钥。这带来两个重要后果：

1. 无需信使或其他安全手段来传输密钥，因为消息可以使用预期接收者公开的加密密钥进行解密。只有他知道相应的解密密钥，因此只有他能解密消息。
2. 可以使用私有的解密密钥对消息进行"签名"。任何人都可以使用相应的公开加密密钥来验证此签名。签名无法伪造，签名者事后也不能否认其签名的有效性。这在"电子邮件"和"电子资金转账"系统中具有明显的应用价值。

加密消息时，将其表示为一个数字 M，将 M 自乘到一个公开指定的幂 e，然后取结果除以公开指定的、由两个大秘密素数 p 和 q 的乘积 n 后的余数。解密过程类似；只是使用另一个不同的秘密幂 d，其中 $e \cdot d \equiv 1 \pmod{(p - 1) \cdot (q - 1)}$。该系统的安全性部分依赖于分解已公开除数 n 的难度。

关键词和短语：数字签名，公钥密码系统，隐私，认证，安全，因式分解，素数，电子邮件，消息传递，电子资金转账，密码学。

CR 分类：2.12, 3.15, 3.50, 3.81, 5.25

I 引言

"电子邮件"[10]的时代可能即将到来；我们必须确保当前"纸质邮件"系统的两个重要特性得以保留：(a) 消息是私密的，(b) 消息可以被签名。我们在本文中演示了如何将这些功能构建到电子邮件系统中。

我们建议的核心是一种新的加密方法。该方法实现了"公钥密码系统"，这是 Diffie 和 Hellman [1] 发明的一个优雅概念。他们的文章启发了我们的研究，因为他们提出了这一概念，但并未提供此类系统的任何实际实现。熟悉 [1] 的读者可能希望直接跳转到第 V 节以了解我们的方法描述。

II 公钥密码系统

在"公钥密码系统"中，每个用户将一个加密过程 E 放入一个公共文件。也就是说，公共文件是一个目录，给出每个用户的加密过程。用户保密其相应解密过程 D 的细节。这些过程具有以下四个属性：

(a) 解密消息 M 的加密形式得到 M。形式化地，

(b) E 和 D 都易于计算。

(c) 通过公开 E，用户并未泄露计算 D 的简便方法。这意味着实际上只有他能解密用 E 加密的消息，或者高效地计算 D。

(d) 如果消息 M 先被解密然后被加密，结果仍是 M。形式化地，

加密（或解密）过程通常由一个通用方法和一个加密密钥组成。在密钥的控制下，通用方法将消息 M 加密，得到消息的加密形式，称为密文 C。每个人都可以使用相同的通用方法；给定过程的安全性将取决于密钥的安全性。揭示加密算法即意味着揭示密钥。

当用户揭示 E 时，他揭示了一种计算 $D(C)$ 的非常低效的方法：测试所有可能的消息 M，直到找到一个使得 $E(M) = C$ 的消息。如果属性 (c) 成立，则需要测试的消息数量将非常庞大，使得这种方法不切实际。

满足 (a)-(c) 的函数 E 是一个"陷门单向函数"；如果它还满足 (d)，则它是一个"陷门单向置换"。Diffie 和 Hellman [1] 引入了陷门单向函数的概念，但未给出任何实例。这些函数被称为"单向"是因为它们在一个方向上易于计算，但在相反方向上（显然）非常难以计算。它们被称为"陷门"函数，因为一旦知道某些私有的"陷门"信息，反函数实际上很容易计算。满足 (d) 的陷门单向函数必须是一个置换：每个消息都是某个其他消息的密文，并且每个密文本本身就是一个允许的消息。（映射是"一对一的"和"满射的"）。属性 (d) 仅是实现"签名"所必需的。

建议读者阅读 Diffie 和 Hellman 的优秀文章 [1] 以获取进一步的背景知识、公钥密码系统概念的详细阐述以及密码学领域其他问题的讨论。公钥密码系统确保隐私和实现"签名"的方式（如下面第 III 和 IV 节所述）也归功于 Diffie 和 Hellman。

在我们的场景中，假设 A 和 B（也称为 Alice 和 Bob）是公钥密码系统的两个用户。我们将用下标区分他们的加密和解密过程：E_A, D_A, E_B, D_B。

III 隐私

加密是使通信私密化的标准手段。发送者在将每个消息传输给接收者之前对其进行加密。接收者（但非未经授权的人）知道适当的解密函数，将其应用于接收到的消息以获得原始消息。窃听者听到传输的消息只听到"乱码"（密文），这对他来说毫无意义，因为他不知道如何解密它。

目前存储在计算机化数据库中并通过电话线传输的大量个人和敏感信息使得加密变得越来越重要。认识到高效、高质量的加密技术需求迫切但供应短缺，美国国家标准局最近采用了在 IBM 开发的"数据加密标准"[13, 14]。新标准不具备实现公钥密码系统所需的属性 (c)。

所有经典加密方法（包括 NBS 标准）都受到"密钥分发问题"的困扰。问题在于，在开始私人通信之前，需要进行另一项私人交易，以将相应的加密和解密密钥分发给发送者和接收者。通常使用私人信使将密钥从发送者运送到接收者。如果电子邮件系统要快速且廉价，这种做法是不可行的。公钥密码系统不需要私人信使；密钥可以通过不安全的通信信道分发。

Bob 如何在公钥密码系统中向 Alice 发送私人消息 M？首先，他从公共文件中检索 E_A。然后他发送给她加密的消息 $E_A(M)$。Alice 通过计算 $D_A(E_A(M)) = M$ 来解密消息。根据公钥密码系统的属性 (c)，只有她能解密 $E_A(M)$。她可以使用同样在公共文件中可用的 E_{B} 来加密私人回复。

请注意，建立私人通信不需要 Alice 和 Bob 之间的任何私人交易。唯一需要的"设置"是，每个希望接收私人通信的用户必须将其加密算法放入公共文件。

两个用户也可以在不查询公共文件的情况下，通过不安全的通信信道建立私人通信。每个用户将自己的加密密钥发送给对方。之后，所有消息都使用接收者的加密密钥进行加密，如同在公钥系统中一样。监听信道的入侵者无法解密任何消息，因为不可能从加密密钥推导出解密密钥。（我们假设入侵者无法修改或向信道中插入消息。）Ralph Merkle 已经开发了该问题的另一种解决方案[5]。

公钥密码系统可用于"引导"进入标准加密方案，例如 NBS 方法。一旦建立了安全通信，传输的第一条消息可以是用于 NBS 方案中的密钥，以编码所有后续消息。如果我们的方法加密速度比标准方案慢，这可能是可取的。（如果使用专用硬件加密设备，NBS 方案可能稍快一些；我们的方案在通用计算机上可能更快，因为多精度算术运算比复杂的位操作更易于实现。）

IV 签名

如果电子邮件系统要取代现有的纸质邮件系统用于商业交易，必须能够对电子消息进行"签名"。签名消息的接收者拥有证据表明消息源自发送者。这一特性强于单纯的认证（接收者可以验证消息来自发送者）；接收者可以说服"法官"签名者发送了该消息。为此，他必须使法官相信他没有自己伪造签名消息！在认证问题中，接收者不担心这种可能性，因为他只想让自己确信消息来自发送者。

电子签名必须是依赖于消息的，也必须是依赖于签名者的。否则，接收者可以在将消息-签名对展示给法官之前修改消息。或者，他可以将签名附加到任何消息上，因为无法检测电子"剪切和粘贴"。

为了实现签名，公钥密码系统必须使用陷门单向置换（即具有属性 (d)）来实现，因为解密算法将应用于未加密的消息。

用户 Bob 如何在公钥密码系统中向 Alice 发送"签名"消息 M？他首先使用 D_B 计算消息 M 的"签名" S：

（根据公钥密码系统的属性 (d)，解密未加密的消息是"有意义的"：每个消息都是某个其他消息的密文。）然后他使用 E_A 加密 S（为了隐私），并将结果 $E_A(S)$ 发送给 Alice。他不需要同时发送 M；可以从 S 计算出来。

Alice 首先用 D_A 解密密文以得到 S。她知道谁是签名的假定发送者（本例中为 Bob）；如果需要，这可以以附加到 S 的明文形式给出。然后她使用发送者的加密过程（本例中为 E_B，可从公共文件获取）提取消息：

她现在拥有一个消息-签名对 (M, S)，其属性类似于签名纸质文档的属性。

Bob 事后不能否认给 Alice 发送过此消息，因为其他任何人都无法创建 $S = D_B(M)$。Alice 可以说服"法官" $E_B(S) = M$，因此她有证据证明 Bob 签署了该文件。

显然，Alice 不能将 M 修改为不同版本 M'，因为那样她也必须创建相应的签名 $S' = D_B(M')$。

因此，Alice 收到了 Bob"签名"的消息，她可以"证明"是他发送的，但她无法修改。（她也不能为任何其他消息伪造他的签名。）

电子支票系统可以基于上述签名系统。不难想象，您家庭终端中的加密设备允许您签署支票，支票通过电子邮件发送给收款人。只需要在每张支票中包含唯一的支票号码，这样即使收款人复制了支票，银行也只兑现它看到的第一张支票。

另一个可能性是，如果加密设备可以做得足够快：将可以进行电话交谈，其中所说的每个词在传输前都由加密设备签名。

当加密如上所述用于签名时，重要的是加密设备不要"硬连线"在终端（或计算机）和通信信道之间，因为消息可能需要使用多个密钥连续解密。也许更自然的是将加密设备视为可以根据需要执行的"硬件子程序"。

我们在上面假设每个用户总是能够可靠地访问公共文件。在"计算机网络"中，这可能很困难；"入侵者"可能伪造据称来自公共文件的消息。用户希望确保他实际上获得的是他期望的通信对象的加密过程，而不是入侵者的加密过程。如果公共文件"签署"它发送给用户的每条消息，这种危险就消失了。用户可以使用公共文件的加密算法 E_{PF} 检查签名。避免在公共文件中"查找" E_{PF} 本身的问题，是通过在用户首次亲自到场加入公钥密码系统并存入其公共加密过程时，给予他 E_{PF} 的描述。然后他存储这个描述，而不是再次查找它。因此，每对用户之间对信使的需求被替换为每个用户与公共文件管理者在加入系统时进行一次安全会面的要求。另一个解决方案是在每个用户注册时，给予他一本书（像电话号码簿），包含系统中所有用户的加密密钥。

V 我们的加密和解密方法

要使用我们的方法加密消息 M，使用公共加密密钥 (e, n)，按以下步骤进行。（这里 e 和 n 是一对正整数。）

首先，将消息表示为介于 0 和 $n - 1$ 之间的整数。（将长消息分成一系列块，并将每个块表示为这样一个整数。）使用任何标准表示。此处的目的不是加密消息，而是使其转换为加密所需的数字形式。

然后，通过将消息自乘到 e 次幂模 n 来加密消息。也就是说，结果（密文 C）是 M^e 除以 n 后的余数。

要解密密文，将其自乘到另一个幂 d，同样模 n。因此，加密和解密算法 E 和 D 如下：

请注意，加密不会增加消息的大小；消息和密文都是 0 到 $n - 1$ 范围内的整数。

因此，加密密钥是正整数对 (e, n)。类似地，解密密钥是正整数对 (d, n)。每个用户公开其加密密钥，并保密相应的解密密钥。（这些整数应该适当地加上下标，如 n_A, e_A 和 d_A，因为每个用户都有自己的集合。但是，我们只考虑一个典型集合，并省略下标。）

如果你想使用我们的方法，应该如何选择加密和解密密钥？

你首先计算 n 作为两个素数 p 和 q 的乘积：

这些素数是非常大的"随机"素数。尽管你将公开 n，但由于分解 n 的巨大难度，因子 p 和 q 将对其他所有人有效地隐藏。这也隐藏了如何从 e 推导出 d 的方法。

然后选择整数 d 为一个与 $(p - 1) \cdot (q - 1)$ 互质的大随机整数。也就是说，检查 d 满足：

（"gcd" 表示 "最大公约数"）。

最后，从 p, q 和 d 计算整数 e，作为 d 模 $(p - 1) \cdot (q - 1)$ 的"乘法逆元"。因此我们有

我们在下一节中证明这保证了 (1) 和 (2) 成立，即 E 和 D 是互逆的置换。第 VII 节展示了如何高效地执行上述每个操作。

上述方法不应与 Diffie 和 Hellman [1] 提出的解决密钥分发问题的"指数运算"技术混淆。他们的技术允许两个用户确定一个用于普通密码系统的共同密钥。它不是基于陷门单向置换。Pohlig 和 Hellman [8] 研究了一种与我们的方案相关的方案，其中指数运算是模一个素数进行的。

VI 基础数学

我们使用一个源于欧拉和费马的恒等式 [7] 来证明解密算法的正确性：对于任何与 n 互质的整数（消息）M，

这里 $\phi(n)$ 是欧拉 totient 函数，给出小于 n 且与 n 互质的正整数的数量。对于素数 p，

在我们的情况下，根据 totient 函数的基本性质 [7]：

由于 d 与 $\phi(n)$ 互质，它在整数模 $\phi(n)$ 环中有一个乘法逆元 e：

我们现在证明方程 (1) 和 (2) 成立（即，如果 e 和 d 按上述方式选择，则解密工作正常）。现在

且

由 (3) 可知，对于所有使得 p 不整除 M 的 M

并且由于 $(p - 1)$ 整除 $\phi (n)$

当 $M \equiv 0 \pmod{p}$ 时，这显然成立，因此这个等式实际上对所有 M 都成立。类似地论证 q 可得

最后这两个方程一起意味着对于所有 M，

这蕴含了对所有 $M, 0 \leq M < n$，(1) 和 (2) 成立。因此 E 和 D 是互逆的置换。（我们感谢 Rich Schroepel 建议了上述对作者先前证明的改进版本。）

VII 算法

为了表明我们的方法是实用的，我们描述每个所需操作的高效算法。

A 如何高效地加密和解密

计算 M^{e} (模 n) 最多需要 $2 \cdot \log_{2}(e)$ 次乘法和 $2 \cdot \log_{2}(e)$ 次除法，使用以下过程（解密可以类似地使用 d 代替 e 来执行）：

步骤 1. 令 $e_k e_{k-1} \ldots e_1 e_0$ 为 e 的二进制表示。

步骤 2. 将变量 C 设为 1。

步骤 3. 对于 $i = k, k - 1, \dots, 0$ 重复步骤 3a 和 3b：

步骤 3a. 将 C 设为 C^2 除以 n 后的余数。

步骤 3b. 如果 $e_i = 1$，则将 C 设为 $C \cdot M$ 除以 n 后的余数。

步骤 4. 停止。现在 C 是 M 的加密形式。

这个过程被称为"通过重复平方和乘法进行指数运算"。这个过程只有最佳方案的一半好；已知有更高效的方案。Knuth [3] 详细研究了这个问题。

加密和解密相同这一事实导致了简单的实现。（整个操作可以在几个专用集成电路芯片上实现。）

一台高速计算机可以在几秒钟内加密一个 200 位的消息 M；专用硬件会快得多。每个块的加密时间增长不快于 n 的位数的立方。

B 如何找到大素数

每个用户必须（私下）选择两个大的随机数 p 和 q 以创建他自己的加密和解密密钥。这些数字必须足够大，以至于任何人从计算上分解 $n = p \cdot q$ 都是不可行的。（记住 n，但不是 p 或 q，将出现在公共文件中。）我们建议使用 100 位（十进制）的素数 p 和 q，这样 n 就有 200 位。

要找到一个 100 位的"随机"素数，生成（奇数）100 位随机数，直到找到一个素数。根据素数定理 [7]，大约需要测试 $(\ln 10^{100}) / 2 = 115$ 个数才能找到一个素数。

为了测试一个大数 b 是否为素数，我们推荐 Solovay 和 Strassen [12] 提出的优雅的"概率"算法。它从 $\{1, \ldots, b - 1\}$ 上的均匀分布中随机选择一个数 a，并测试是否

其中 $J(a, b)$ 是雅可比符号 [7]。如果 b 是素数，则 (6) 总是成立。如果 b 是合数，则 (6) 为假的概率至少为 $1/2$。如果 (6) 对于 100 个随机选择的 a 值都成立，那么 b 几乎肯定是素数；存在 2^{100} 分之一的（可忽略的）概率 b 是合数。即使偶然在我们的系统中使用了合数，接收者可能会通过注意到解密无法正常工作来检测到这一点。当 b

是奇数，$a \leq b$，且 $\gcd(a, b) = 1$ 时，雅可比符号 $J(a, b)$ 的值在 $\{-1, 1\}$ 中，并且可以通过以下程序高效计算：

（$J(a, b)$ 和 $\gcd(a, b)$ 的计算也可以很好地结合起来。）请注意，该算法不是通过尝试分解一个数来测试其是否为素数。测试大数素性的其他高效程序在 [6,9,11] 中给出。

为了获得针对复杂分解算法的额外保护，p 和 q 的长度应相差几位，$(p - 1)$ 和 $(q - 1)$ 都应包含大素因子，并且 $\gcd(p - 1, q - 1)$ 应该很小。后一个条件很容易检查。

要找到一个素数 p 使得 $(p - 1)$ 有一个大素因子，首先生成一个大的随机素数 u，然后令 p 为序列 $i \cdot u + 1$（$i = 2, 4, 6, \ldots$）中的第一个素数。（这应该不需要太长时间。）通过确保 $(u - 1)$ 也有一个大素因子，可以提供额外的安全性。

一台高速计算机可以在几秒钟内确定一个 100 位数是否为素数，并且可以在一两分钟内找到给定点之后的第一个素数。

寻找大素数的另一种方法是取一个已知因式分解的数，对其加一，然后测试结果是否为素数。如果找到了一个素数 p，可以通过使用 $p - 1$ 的因式分解来证明它确实是素数。我们省略对此的讨论，因为概率方法已经足够。

C 如何选择 d

选择一个与 $\phi(n)$ 互质的数 d 非常容易。例如，任何大于 $\max(p, q)$ 的素数都可以。重要的是 d 应该从一个足够大的集合中选择，以使密码分析者无法通过直接搜索找到它。

D 如何从 d 和 $\phi(n)$ 计算 e

要计算 e，请使用欧几里得算法的以下变体来计算 $\phi(n)$ 和 d 的最大公约数。（参见 [3] 中的练习 4.5.2.15。）通过计算序列 $x_0, x_1, x_2, \ldots$ 来计算 $\gcd(\phi(n), d)$，其中 $x_0 \equiv \phi(n)$，$x_1 = d$，且 $x_{i+1} \equiv x_{i-1} \pmod{x_i}$，直到找到等于 0 的 x_k。然后 $\gcd(x_0, x_1) = x_{k-1}$。为每个 x_i 计算满足 $x_i = a_i \cdot x_0 + b_i \cdot x_1$ 的数 a_i 和 b_i。如果 $x_{k-1} = 1$，那么 b_{k-1} 是 x_1 模 x_0 的乘法逆元。由于 k 将小于 $2\log_2(n)$，这个计算非常快。

如果 e 结果小于 $\log_2(n)$，则通过选择另一个 d 值重新开始。这保证了每个加密消息（除了 $M = 0$ 或 $M = 1$）都会经历某种"回绕"（模 n 约简）。

VIII 一个小例子

考虑 $p = 47, q = 59, n = p \cdot q = 47 \cdot 59 = 2773$，且 $d = 157$ 的情况。那么 $\phi(2773) = 46 \cdot 58 = 2668$，可以如下计算 e：

因此 $e = 17$，是 $d = 157$ 模 2668 的乘法逆元。

使用 $n = 2773$，我们可以每块编码两个字母，用两位数字代替每个字母：空格 $= 00$，A $= 01$，B $= 02$，...，Z $= 26$。因此消息

ITS ALL GREEK TO ME

（Julius Caesar, I, ii, 288, 释义）被编码为：

0920 1900 0112 1200 0718 0505 1100 2015 0013 0500

由于 $e = 10001$ 是二进制，第一块（$M = 920$）被加密为：

整个消息被加密为：

0948 2342 1084 1444 2663 2390 0778 0774 0219 1655 .

读者可以验证解密有效：$948^{157} \equiv 920 \pmod{2773}$，等等。

IX 方法的安全性：密码分析方法

由于不存在技术来证明加密方案是安全的，唯一可用的测试是看是否有人能想出破解它的方法。NBS 标准就是通过这种方式"认证"的；IBM 花费了 17 人年的时间试图破解该方案，但徒劳无功。一旦一个方法成功地抵抗了这样的集中攻击，出于实际目的可以认为它是安全的。（实际上，关于 NBS 方法的安全性存在一些争议 [2]。）

我们在接下来的章节中表明，破解我们系统的所有明显方法至少与分解 n 一样困难。虽然分解大数并未被证明是困难的，但它是一个众所周知的问题，过去三百年中许多著名数学家都研究过它。费马 (1601?-1665) 和勒让德 (1752-1833) 发展了分解算法；今天一些更高效的算法是基于勒让德的工作。然而，正如我们将在下一节中看到的，尚未有人找到一种算法可以在合理的时间内分解一个 200 位的数字。我们得出结论，我们的系统已经通过这些先前寻找高效分解算法的努力得到了部分的"认证"。

在接下来的章节中，我们考虑密码分析者可能尝试从公开的加密密钥确定秘密解密密钥的方式。我们不考虑保护解密密钥免遭盗窃的方法；通常的物理安全方法应该足够了。（例如，加密设备可以是一个单独的设备，也可以用于生成加密和解密密钥，使得解密密钥永远不会被打印出来（即使对其所有者），而仅用于解密消息。如果设备被篡改，它可以擦除解密密钥。）

A 分解 n

分解 n 将使敌方密码分析者能够"破解"我们的方法。n 的因子使他能够计算 $\phi(n)$，从而计算 d。幸运的是，分解一个数字似乎比确定它是素数还是合数要困难得多。

存在大量的分解算法。Knuth [3, Section 4.5.4] 给出了其中许多算法的精彩介绍。Pollard [9] 提出了一种算法，可以在 $O(n^{1/4})$ 的时间内分解一个数 n。

作者所知的最快分解算法归功于 Richard Schroepel（未发表）；它可以在大约

步（这里 $\ln$ 表示自然对数函数）内分解 n。表 1 给出了使用 Schroeppel 方法分解 n 所需的操作次数，以及如果每次操作使用一微秒所需的时间，针对不同长度（十进制位数）的数字 n。

表 1

位数	操作次数	时间
50	1.4×10^{10}	3.9 小时
75	9.0×10^{12}	104 天
100	2.3×10^{15}	74 年
200	1.2×10^{23}	3.8×10^{9} 年
300	1.5×10^{29}	4.9×10^{15} 年
500	1.3×10^{39}	4.2×10^{25} 年

我们建议 n 的长度约为 200 位。根据具体应用中加密速度和安全性的相对重要性，可以使用更长或更短的长度。80 位的 n 对使用当前技术的攻击提供了中等安全性；使用 200 位则为未来的发展提供了安全余地。这种选择密钥长度（从而选择安全级别）的灵活性是许多先前加密方案（如 NBS 方案）所不具备的特性。

B 不分解 n 计算 $\phi(n)$

如果密码分析者能够计算 $\phi(n)$，那么他可以通过计算 e 模 $\phi(n)$ 的乘法逆元作为 d（使用第 VII 节 D 部分的过程）来破解系统。

我们认为这种方法并不比分解 n 更容易，因为它使密码分析者能够轻松地使用 $\phi(n)$ 分解 n。这种分解 n 的方法尚未被证明是实用的。

如何使用 $\phi(n)$ 分解 n？首先，从 n 和 $\phi(n) = n - (p + q) + 1$ 得到 $(p + q)$。然后 $(p - q)$ 是 $(p + q)^2 - 4n$ 的平方根。最后，q 是 $(p + q)$ 和 $(p - q)$ 的差的一半。

因此，通过计算 $\phi(n)$ 来破解我们的系统并不比通过分解 n 来破解我们的系统更容易。（这就是为什么 n 必须是合数；如果 n 是素数，计算 $\phi(n)$ 是微不足道的。）

C 不分解 n 或计算 $\phi(n)$ 确定 d

当然，d 应该从一个足够大的集合中选择，以便对其进行直接搜索是不可行的。

我们认为对于密码分析者来说，计算 d 并不比分解 n 更容易，因为一旦知道 d，就可以轻松分解 n。这种分解方法也尚未被证明是富有成果的。

知道 d 后可以如下分解 n。一旦密码分析者知道 d，他就可以计算 $e \cdot d - 1$，这是 $\phi(n)$ 的一个倍数。Miller [6] 已经证明，可以使用 $\phi(n)$ 的任何倍数来分解 n。因此，如果 n 很大，密码分析者不应该能够比分解 n 更容易地确定 d。

密码分析者可能希望找到一个 d'，它等同于公钥密码系统用户秘密持有的 d。如果这样的 d' 值很常见，那么暴力搜索可能会破解该系统。然而，所有这样的 d' 相差 $(p - 1)$ 和 $(q - 1)$ 的最小公倍数，并且找到其中一个就能分解 n。（在 (3) 和 (5) 中，$\phi(n)$ 可以替换为 $\operatorname{lcm}(p - 1, q - 1)$。）因此，找到任何这样的 d' 与分解 n 一样困难。

D 以其他方式计算 D

尽管"不分解 n 计算模 n 的 e 次方根"这个问题不像分解那样是一个众所周知的困难问题，但我们相当有信心它在计算上是难以处理的。有可能证明任何破解我们方案的通用方法都会产生一个高效的分解算法。这将确立任何破解我们方案的方法必须与分解一样困难。然而，我们尚未能证明这个猜想。

我们的方法应该通过让上述难处理性猜想经受住集中试图反驳它的努力来得到认证。我们挑战读者找到一种"破解"我们方法的方式。

X 加密签名消息时避免"重新分块"

签名消息可能需要进行"重新分块"以便加密，因为签名 n 可能大于加密 n（每个用户都有自己的 n）。这可以避免如下。为公钥密码系统选择一个阈值 h（比如 $h = 10^{199}$）。每个用户维护两个公共的 (e, n) 对，一个用于加密，一个用于签名验证，其中每个签名 n 都小于 h，每个加密 n 都大于 h。然后，加密签名消息就不需要重新分块；消息根据发送者的签名 n 进行分块。

另一个解决方案使用了 [4] 中给出的一种技术。每个用户有一个单一的 (e,n) 对，其中 n 在 h 和 $2h$ 之间，h 是如上的阈值。消息被编码为一个小于 h 的数字，并像以前一样加密，除非密文大于 h，在这种情况下它会重复加密直到小于 h。类似地，对于解密，密文被重复解密以获得一个小于 h 的值。如果 n 接近 h，重新加密将很少发生。（无限循环是不可能的，因为最坏情况下消息被加密为其本身。）

XI 结论

我们提出了一种实现公钥密码系统的方法，其安全性部分依赖于分解大数的难度。如果我们方法的安全性被证明是足够的，它允许在不使用信使携带密钥的情况下建立安全通信，并且还允许对数字化文档进行"签名"。

该系统的安全性需要更详细地研究。特别是，分解大数的难度应该非常仔细地研究。我们敦促读者找到一种"破解"该系统的方法。一旦该方法在足够长的时间内经受住了所有攻击，它就可以以合理的信心程度使用。

我们的加密函数是作者所知的"陷门单向置换"的唯一候选者。可能需要找到其他例子，以便如果我们的系统安全性在未来的某一天不足时，可以提供替代实现。这些函数肯定还有许多新的应用有待发现。

致谢。我们感谢 Martin Hellman、Richard Schroepel、Abraham Lempel 和 Roger Needham 的有益讨论，以及 Wendy Glasser 在准备初稿时的协助。Xerox PARC 为准备最终手稿提供了支持和一些极好的文本编辑设施。

收到日期：1977年4月4日；修订日期：1977年9月1日。

参考文献

1. Diffie, W., and Hellman, M. New directions in cryptography. IEEE Trans. Inform. Theory IT-22, (Nov. 1976), 644-654.
2. Diffie, W., and Hellman, M. Exhaustive cryptanalysis of the NBS data encryption standard. Computer 10 (June 1977), 74-84.
3. Knuth, D. E. The Art of Computer Programming, Vol 2: Seminumerical Algorithms. Addison-Wesley, Reading, Mass., 1969.
4. Levine, J., and Brawley, J.V. Some cryptographic applications of permutation polynomials. Cryptologia 1 (Jan. 1977), 76-92.
5. Merkle, R. Secure communications over an insecure channel. Submitted to Comm. ACM.
6. Miller, G.L. Riemann's hypothesis and tests for primality. Proc. Seventh Annual ACM Symp. on the Theory of Comptng. Albuquerque, New Mex., May 1975, pp. 234-239; extended vers. available as Res. Rep. CS-75-27, Dept. of Comptr. Sci., U. of Waterloo, Waterloo, Ont., Canada, Oct. 1975.
7. Niven, I., and Zuckerman, H.S. An Introduction to the Theory of Numbers. Wiley, New York, 1972.
8. Pohlig, S.C., and Hellman, M.E. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. To appear in IEEE Trans. Inform. Theory, 1978.
9. Pollard, J.M. Theorems on factorization and primality testing. Proc. Camb. Phil. Soc. 76 (1974), 521-528.
10. Potter, R.J., Electronic mail. Science 195, 4283 (March 1977), 1160-1164.
11. Rabin, M.O., Probabilistic algorithms. In Algorithms and Complexity, J. F. Traub, Ed., Academic Press, New York, 1976, pp. 21-40.
12. Solovay, R., and Strassen, V. A Fast Monte-Carlo test for primality. SIAM J. Comptng. (March 1977), 84-85.
13. Federal Register, Vol. 40, No. 52, March 17, 1975.
14. Federal Register, Vol. 40, No. 149, August 1, 1975.

专业术语中英文对照表

英文术语	中文翻译
digital signatures	数字签名
public-key cryptosystems	公钥密码系统
privacy	隐私
authentication	认证
security	安全性
factorization	因式分解
prime number	素数

英文术语	中文翻译
electronic mail	电子邮件
message-passing	消息传递
electronic funds transfer	电子资金转账
cryptography	密码学
encryption method	加密方法
encryption key	加密密钥
decryption key	解密密钥
public file	公共文件
trap-door one-way function	陷门单向函数
trap-door one-way permutation	陷门单向置换
one-way function	单向函数
ciphertext	密文
modular arithmetic	模运算
multiplicative inverse	乘法逆元
Euler's totient function	欧拉 totient 函数
greatest common divisor (gcd)	最大公约数
probabilistic algorithm	概率算法
primality testing	素性测试
Jacobi symbol	雅可比符号
exponentiation by repeated squaring and multiplication	通过重复平方和乘法进行指数运算
key distribution problem	密钥分发问题
National Bureau of Standards (NBS)	国家标准局
Data Encryption Standard (DES)	数据加密标准
cryptanalyst	密码分析者
reblocking	重新分块
threshold value	阈值
computational intractability	计算难处理性