现代密码学 COMP401227

第 6 章：数字签名

赵俊舟

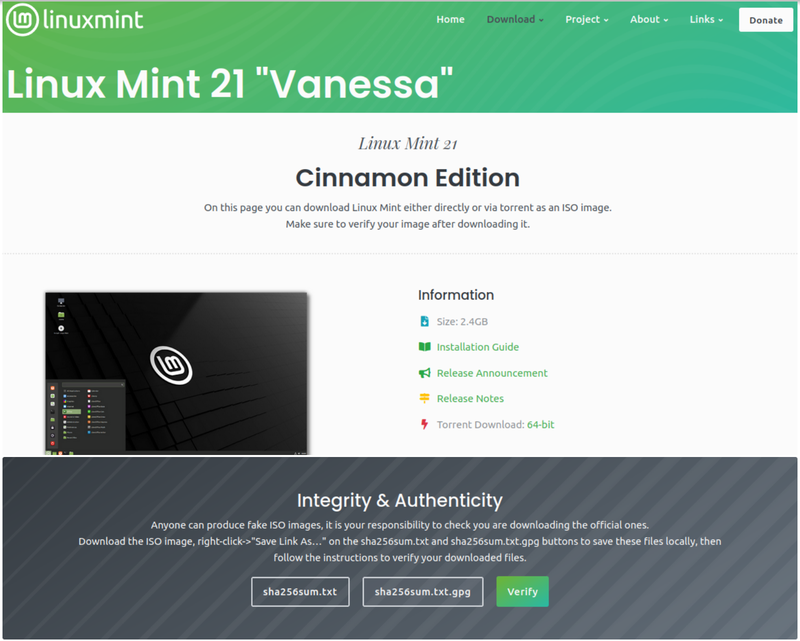
junzhou.zhao@xjtu.edu.cn

2025 年 4 月 9 日

目录

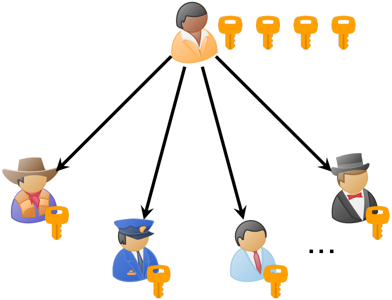
目录

消息认证码的问题


linuxmint
Home
Download
Project
About
Links
Donate
Linux Mint 21 "Vanessa"
Linux Mint 21
Cinnamon Edition
On this page you can download Linux Mint either directly or via torrent as an ISO image.
Make sure to verify your image after downloading it.
Information
Size: 2.4GB
Installation Guide
Release Announcement
Release Notes
Torrent Download: 64-bit
Integrity & Authenticity
Anyone can produce fake ISO images, it is your responsibility to check you are downloading the official ones.
Download the ISO image, right-click->"Save Link As..." on the sha256sum.txt and sha256sum.txt.gpg buttons to save these files locally, then follow the instructions to verify your downloaded files.
sha256sum.txt
sha256sum.txt.gpg
Verify

消息认证码的问题：密钥管理问题

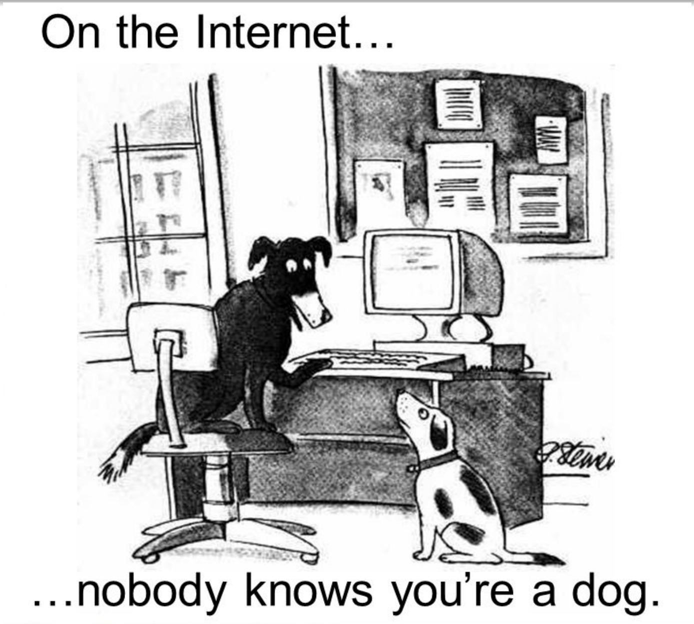
- 消息认证码可以对信源的正确性和消息的完整性进行验证。
- 消息认证码是基于对称密码实现的，如果 n 个接收方（例如软件下载用户）要对发送方发送的消息（例如下载的软件）进行验证，那么需要在发送方和接收方之间管理 n 个密钥。
- 这样会对发送方带来很大的麻烦。例如，某些热门软件有很大的下载量，软件发布方需要管理大量密钥。



消息认证码的问题：假冒和否认问题

- 消息认证可以保护信息交换双方不受第三方攻击，但是它不能处理通信双方自身发生的攻击。
- 例如，当 Alice 给 Bob 发送一条认证消息时：
- 假冒问题：Bob 可以伪造一条消息并声称该消息发自 Alice。Bob 只需要生成一条消息，并用 Alice 和 Bob 共享的密钥产生认证码，并将认证码附于消息之后。
- 否认问题：Alice 可以否认曾经给 Bob 发送过消息。因为 Bob 可以伪造消息，所以无法证明 Alice 确实发送过该消息。
- 在收发双方不能完全信任的情况下，需要其他方法来解决这些问题——数字签名（Digital Signature）。

数字签名



数字签名

数字签名

使以数字形式存储的明文信息经过特定密码变换生成密文，作为相应明文的签名，使明文信息的接收者能够验证信息确实来自合法用户，以及确认信息发送者身份。

对数字签名的基本要求：

- 签名必须是与消息相关的二进制串。
- 签名必须使用发送方某些独有的信息，以防伪造和否认。
- 产生数字签名比较容易。
- 识别和验证数字签名比较容易。
- 伪造数字签名在计算上不可行。
- 保存数字签名的副本是可行的。

数字签名的基本形式

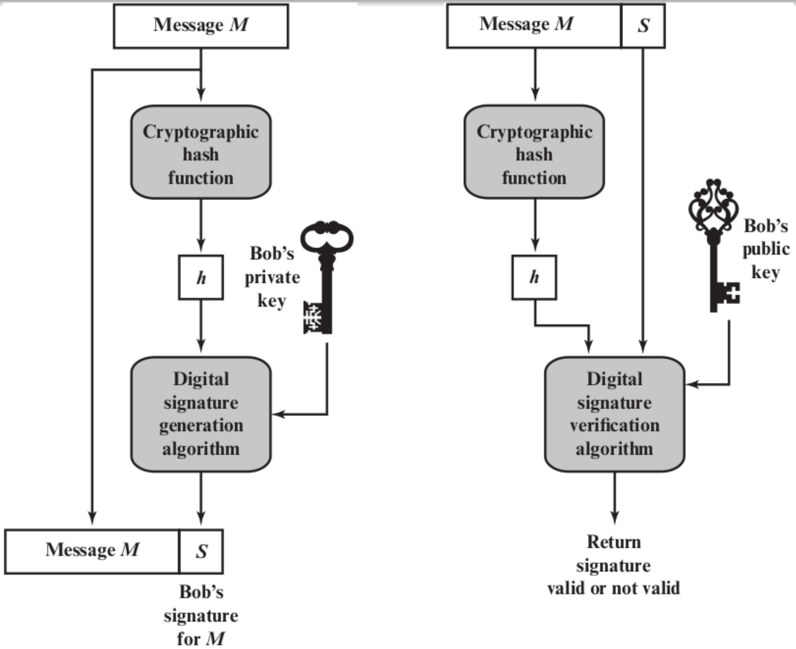
数字签名的方式:

- 对消息整体的签名：将被签消息整体经过密码变换得到签名。
- 对消息摘要的签名：先计算消息摘要，再对消息摘要进行密码变换得到签名。

两类数字签名:

- 确定性数字签名：明文与签名一一对应。
- 概率性数字签名：一个明文可以有多个合法签名，每次都不一样。

数字签名的一般模型


Message M
Cryptographic hash function
h
Bob's private key
Digital signature generation algorithm
Message M
S
Bob's signature for M
Message M
S
Cryptographic hash function
h
Bob's public key
Digital signature verification algorithm
Return signature valid or not valid

目录

ElGamal 密码体系及数字签名

- 1985 年，Taher ElGamal 提出了一种基于离散对数的公开密钥体制，与 Diffie-Hellman 密钥分配体制密切相关。
- ElGamal 密码体制应用于数字签名标准和 S/MIME 电子邮件标准。

初始化

- Alice 和 Bob 共享大素数 p 及其本原元 g。
- Alice 和 Bob 分别选择私钥 x_A 和 x_B，并计算各自的公钥 $y_A = g^{x_A} \bmod p$ 和 $y_B = g^{x_B} \bmod p$。

ElGamal 密码体系

加密

- Alice 选择任意整数 $r \in \mathbb{Z}_p$，并计算 $k = {y_B}^r \bmod p$
- 将明文 m 加密为密文 $c = (c_1, c_2)$，其中 $c_1 = g^r \bmod p$，$c_2 = km \bmod p$

解密

- Bob 首先恢复 $k = c_1^{x_B} \bmod p = g^{rx_B} \bmod p = y_B^r \bmod p$
- 然后恢复明文 $m = c_2 k^{-1} \bmod p$

等价于：Alice 每次发送消息时选择了一个临时私钥 r，c_1 为该临时私钥对应的公钥，k 为协商的密钥。

ElGamal 数字签名方案

Alice 签名

- 计算哈希值 $h = H(m) \in \mathbb{Z}_p$
- 选择任意整数 $r \in \mathbb{Z}_{p-1}^*$
- 计算 $s_1 = g^r \bmod p$
- 计算 $s_2 = r^{-1}(h - x_A s_1) \bmod (p-1)$
- 得到签名 (s_1, s_2)

Bob 验证

- 计算 $v_1 = g^h \bmod p$，计算 $v_2 = {y_A}^{s_1} s_1^{s_2} \bmod p$
- 如果 $v_1 = v_2$，则签名合法；否则签名不合法。

ElGamal 数字签名方案

$$
\begin{aligned}
v_1 &= v_2 \\
g^h &\equiv y_A^{s_1} s_1^{s_2} \bmod p \\
g^h &\equiv g^{x_A s_1} g^{r s_2} \bmod p \\
g^{h - x_A s_1} &\equiv g^{r s_2} \bmod p \\
h - x_A s_1 &\equiv r s_2 \pmod{(p-1)} \\
h - x_A s_1 &\equiv r r^{-1}(h - x_A s_1) \pmod{(p-1)} \\
h - x_A s_1 &\equiv (h - x_A s_1) \pmod{(p-1)}
\end{aligned}
$$

注意第五个等号成立是因为

$$g^i \equiv g^j \pmod{p} \iff i \equiv j \pmod{(p-1)}$$

ElGamal 数字签名举例

例 (Alice 产生密钥对)

- 对整数域 $\mathrm{GF}(19)$，即 $p = 19$，选择素根 $g = 10$；
- Alice 选择私钥 $x_A = 16$，则公钥 $y_A = g^{x_A} \bmod p = 4$。

例 (假设 Alice 要对哈希值为 14 的消息进行签名)

- 选择 $r = 5$，满足 $\gcd(r, p-1) = 1$ 且 $r^{-1} \bmod (p-1) = 11$；
- $s_1 = g^r \bmod p = 3$；
- $s_2 = r^{-1}(h - x_A s_1) \bmod (p-1) = 4$。

例 (Bob 验证签名)

- $v_1 = g^h \bmod p = 16$, $v_2 = y_A^{s_1} s_1^{s_2} \bmod p = 16$；
- 因为 $v_1 = v_2$，所以签名合法。

目录

Schnorr 数字签名方案

- Schnorr 签名算法由德国数学家、密码学家克劳斯 · 施诺于 1990 年提出。
- Schnorr 签名的特点：计算简便，生成签名的主要工作不依赖于消息，可以在处理器空闲时间完成。

初始化

- 选择大素数 p，使得 $p-1$ 包含大素数因子 q;
- 选择整数 g，使得 $g^q \equiv 1 \pmod{p}$;
- g, p 和 q 公开，作为全局公钥参数;
- 私钥：随机选择整数 $x \in \mathbb{Z}_q$ 作为私钥;
- 公钥：计算 $y = g^x \bmod p$ 作为公钥。

Schnorr 数字签名方案

签名

- 选择随机整数 $k \in \mathbb{Z}_p$，并计算 $r = g^k \bmod p$；
- 将 r 附在消息 m 后面一起计算哈希值 $e = H(m\|r)$；
- 计算 $s = (k + xe) \bmod q$，得到签名 (e, s)。

验证

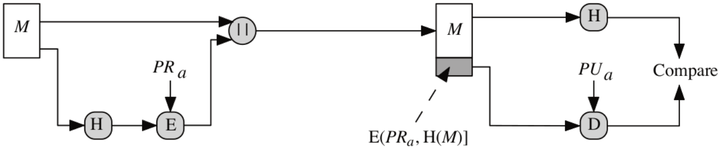
- 计算 $r' = g^s y^{-e} \bmod p$；
- 验证是否 $e \stackrel{?}{=} H(m\|r')$。只需验证 $r = r'$ 是否成立：
$$r' \equiv g^s y^{-e} \equiv g^s g^{-xe} \equiv g^{s-xe} \equiv g^k \equiv r \pmod p$$

目录

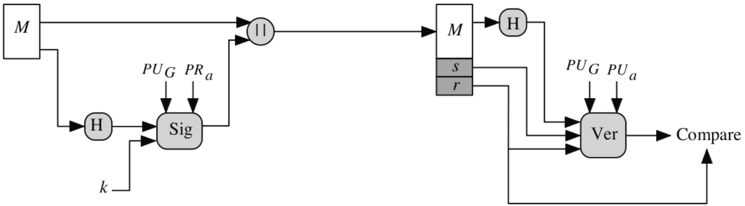
数字签名标准（Digital Signature Standard, DSS）

- DSS 是 NIST 作为联邦信息处理标准 FIPS 186 发布的；
- 由 NIST 和 NSA 在 90 年代早期设计；
- DSS 是标准，DSA 是其算法；
- DSS 是 ElGamal 和 Schnorr 算法的变形；
- DSS 使用 SHA 作为哈希算法；
- DSS 产生 320 位数字签名，但是具有 512-1024 位的安全性；
- DSS 的安全依赖于 DLP 问题。

DSA vs. RSA



(a) RSA Approach



(b) DSA Approach

与 RSA 相比，DSA 只提供数字签名，不能用于加密或密钥交换。

DSA 密钥生成

全局共享参数 p, q, g

- p： L 位大素数，其中 $512 \leq L \leq 1024$，是 64 整倍数；
- q： $p-1$ 包含素因子 q，长度 N 位（例如 $N = 160$）；
- g： 选择整数 g，使得 $g^q \equiv 1 \pmod{p}$。

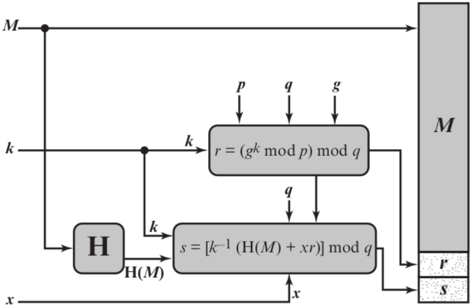
产生用户密钥对

- 用户选择私钥 $x \in \mathbb{Z}_q$
- 用户计算公钥 $y = g^x \bmod p$

DSA 签名的产生

对消息 m 签名

- 计算消息摘要 $h = H(m)$
- 产生随机数 $k \in \mathbb{Z}_q^*$
- 计算 $r = g^k \bmod p \bmod q$
- 计算 $s = k^{-1}(h + xr) \bmod q$

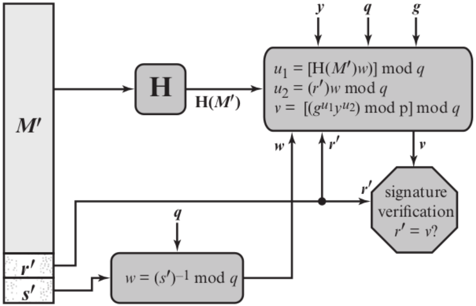


把消息 m 和签名 (r, s) 一起发送给接收方。

DSA 签名的验证

接收方验证消息 m 和签名 (r, s)

- 计算消息摘要 $h = H(m)$
- 计算辅助值 $w = s^{-1} \bmod q$
- 计算辅助值 $u_1 = hw \bmod q$
- 计算辅助值 $u_2 = rw \bmod q$
- 计算 $v = g^{u_1} y^{u_2} \bmod p \bmod q$



如果 $v = r$，则签名合法。

DSA 签名正确性推导

- 因为 $\{g^i \bmod p\}$ 的最小周期为 q，即 $g^i \equiv g^{i \bmod q} \bmod p$。
- 所以

$$
\begin{aligned}
v &= g^{u_1} y^{u_2} \bmod p \bmod q \\
&= g^{u_1} g^{xu_2} \bmod p \bmod q \\
&= g^{hw} g^{xrw} \bmod p \bmod q \\
&= g^{hw+xrw} \bmod p \bmod q
\end{aligned}
$$

- 又因为 $s = k^{-1}(h + xr) \bmod q$ 且 $w = s^{-1} \bmod q$，所以
$$k \equiv s^{-1}(h + xr) \equiv w(h + xr) \equiv (hw + xrw) \pmod q$$
- 所以
$$r = g^k \bmod p \bmod q = g^{hw+xrw} \bmod p \bmod q$$
- 当 $v = r$ 时，可以说明签名是合法的。

DSA

Global Public-Key Components

p prime number where $2^{L-1} < p < 2^L$
for $512 \le L \le 1024$ and L a multiple of 64;
i.e., bit length L between 512 and 1024 bits
in increments of 64 bits

q prime divisor of $(p-1)$, where $2^{N-1} < q < 2^N$
i.e., bit length of N bits

g $= h(p-1)/q$ is an exponent mod p,
where h is any integer with $1 < h < (p-1)$
such that $h^{(p-1)/q} \bmod p > 1$

User's Private Key

x random or pseudorandom integer with $0 < x < q$

User's Public Key

$y = g^x \bmod p$

User's Per-Message Secret Number

k random or pseudorandom integer with $0 < k < q$

Signing

$r = (g^k \bmod p) \bmod q$

$s = [k^{-1}(H(M) + xr)] \bmod q$

Signature $= (r, s)$

Verifying

$w = (s')^{-1} \bmod q$

$u_1 = [H(M')w] \bmod q$

$u_2 = (r')w \bmod q$

$v = [(g^{u1}y^{u2}) \bmod p] \bmod q$

TEST: $v = r'$

M = message to be signed
H(M) = hash of M using SHA-1
M', r', s' = received versions of M, r, s

目录

椭圆曲线数字签名（ECDSA）

- 2009 年修订的 FIPS 186 加入了椭圆曲线数字签名算法；
- ECDSA 密码效率较高，可以使用较短密钥，ECDSA 越来越流行；
- ECDSA 主要包括四个部分：
  - 确定全局参数，包括椭圆曲线参数及基准点；
  - 签名者产生密钥对；
  - 对消息产生哈希值，签名者使用私钥、全局参数、哈希值生成签名；
  - 验证者使用签名者公钥、全局参数验证签名是否合法。

ECDSA 全局参数及密钥产生

以 $\mathrm{GF}(p)$ 上的素数域椭圆曲线为例

生成全局参数

- 随机选择大素数 p；
- a, b 为椭圆曲线参数；
- G 为基准点；
- n 为点 G 的阶，即满足 $nG = O$ 的最小正整数。

生成公私钥对

- 选择随机整数 $d \in \mathbb{Z}_n$ 作为私钥；
- 计算公钥 $Q = dG$。

ECDSA 数字签名的产生

为消息 m 产生签名

1. 计算消息摘要 $h = H(m)$
2. 选择随机整数 $k \in \mathbb{Z}_n^*$
3. 计算 $(x, y) = kG$，以及 $r = x \bmod n$
4. 计算 $s = k^{-1}(h + dr) \bmod n$
5. 消息 m 的签名为 (r, s)。

ECDSA 数字签名的验证

验证消息 m 的签名 (r, s) 是否合法

1. 计算消息摘要 $h = H(m)$
2. 计算辅助值 $w = s^{-1} \bmod n$
3. 计算辅助值 $u_1 = hw$ 和 $u_2 = rw$
4. 计算 $(x_1, y_1) = u_1 G + u_2 Q$
5. 计算 $v = x_1 \bmod n$
6. 当 $v = r$ 时，接受该签名。

ECDSA 数字签名的验证

- 因为
  $u_1G+u_2Q=u_1G+u_2dG=(u_1+u_2d)G=\big((u_1+u_2d) \bmod n\big)G$
- 又因为 $s=k^{-1}(h+dr) \bmod n$，所以
$$\begin{aligned} k &= s^{-1}(h+dr) \bmod n \\ &= w(h+dr) \bmod n \\ &= (hw+rwd) \bmod n \\ &= (u_1+u_2d) \bmod n \end{aligned}$$
- 因此 $u_1G+u_2Q=kG$。
- 在验证时，有 $v=x_1 \bmod n$，其中 $(x_1,y_1)=u_1G+u_2Q=kG$
- 在签名时，有 $r=x \bmod n$，其中 $(x,y)=kG$
- 故当 $v=r$ 时，签名合法。

数字签名标准现状（2023 年 2 月）

- NIST is publishing a revised DSS (FIPS 186-5) and Recommendations for Discrete Logarithm-based Cryptography: Elliptic Curve Domain Parameters (NIST SP 800-186)[1].
- FIPS 186-5 specifies three techniques for the generation and verification of digital signatures:
  - Rivest-Shamir-Adleman (RSA) Algorithm
  - Elliptic Curve Digital Signature Algorithm (ECDSA)
  - Edwards Curve Digital Signature Algorithm (EdDSA)
- DSA, which was specified in prior versions of FIPS 186, is retained only for the purposes of verifying existing signatures.

[1] https://www.nist.gov/news-events/news/2023/02/nist-revises-digital-signature-standard-dss-and-publishes-guideline

小结

1. 数字签名概述
2. ElGamal 数字签名方案
3. Schnorr 数字签名方案
4. 数字签名标准 DSS
5. 椭圆曲线数字签名标准 ECDSA